THE QUANTIFICATION OF INFORMATION SYSTEMS RISK

A LOOK AT QUANTITATIVE RESPONSES TO INFORMATION SECURITY ISSUES

by

Craig S. Wright

A Thesis submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

Charles Sturt University

2017

Abstract

This thesis demonstrates information security can be modelled through a systematic integration of the human, system and software aspects of risk. The creation of risk models based on the deployment of a combination of these approaches drawing on the advanced statistical techniques now available and the creation of game theoretic quantitative models of risk to information systems within set confidence levels is shown to be achievable. This research demonstrates that it is feasible to investigate and quantify the root cause of security flaws that act as a source of system compromise allowing business and governments to most efficiently allocate funds in controlling risk. The thesis demonstrates that to do this requires integrated models that account for the various risk dimensions in information security. Research into the effects of poor system design, market-based risk solutions based on derivative instruments and the impact of common system misconfigurations is incorporated into multivariate survival models. This research also addresses the economic impact of various decisions as a means of determining the optimal distribution of costs and liability when applied to information security and when assigning costs in computer system security and reliability engineering.

Acknowledgments

The author wishes to express sincere appreciation to my principal and second supervisors Dr. Tanveer Zia, who has helped guide me through this research effort, and to Dr. Alfred Wong, for his invaluable input.

I thank my family, friends and co-workers, all of whom have demonstrated infinite patience throughout this effort.

Most importantly I would like to offer my sincerest thanks to Ms. Ramona Watts for her patience, dedication and care in offering support when it was needed most.

Finally, but far from least, I would like to thank the many anonymous reviewers who have together made this thesis possible. Without their comments, critique and suggestions, the material here presented would not be complete.

Professional editor, Dr Gaye Wilson, provided copyediting and proofreading services, per the guidelines laid out in the university-endorsed national guidelines, 'The editing of research theses by professional editors'. All remaining errors are my own.

Professional Editorial Assistance

Editing firm, Editage (www.editage.com) provided proofing services and updated the graphs and figures to ensure that all graphs and figures are sufficiently legible by either increasing the size of the font or the image file.

Professional editor, Dr Gaye Wilson, provided copyediting and proofreading services, per the guidelines laid out in the university-endorsed national guidelines, 'The editing of research theses by professional editors'.

All editing work was limited to formatting, grammar and style.

Table of Contents

Certificate of Authorship

I hereby declare that this submission is my own work and to the best of my knowledge and belief, understand that it contains no material previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any other degree or diploma at Charles Sturt University or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by colleagues with whom I have worked at Charles Sturt University or elsewhere during my candidature is fully acknowledged.

I agree that this thesis be accessible for the purpose of study and research in accordance with normal conditions established by the Executive Director, Library Services, Charles Sturt University or nominee, for the care, loan and reproduction of thesis, subject to confidentiality provisions as approved by the University.

Name: Craig Steven Wright

Date: 08 Feb 2017

I List of figures

II List of Tables

III Published papers in this thesis

Each of the papers used in this thesis is published in a peer reviewed conference proceeding. The trade-off for this approach is that the flow of the thesis becomes more disjointed as the chapters consist of a series of separate publications which have been joined to form a body of research. Each paper has been presented in the original format (excepting that appendixes have been removed to the end of the thesis and that the abstract has been incorporated into the main chapter body). Each peer-reviewed paper has been updated to allow publication per the reviewer's requirements and to make a more coherent flow.

The other concern with this approach is that some papers duplicate the material presented in other published papers. The result is that the reader will see the same argument presented in multiple sections. Leaving this was a difficult decision, but the alternative would have been to edit the published papers and hence to change the nature of the thesis from the presentation of a series of peer-reviewed documents into a completely new section based on the former publication.

The papers presented in this thesis (by chapter) are listed below.

The publications include the following ERA statistics:

- One **ERA A** conference paper,
- Six **ERA B**, conference papers.

Chapter 3

1. "The Economics of Developing Security Embedded Software", (Wright & Zia, 2010). Presented at the 8th Australian Information Security Management Conference (SecAU 2010). **ERA B** Conference. (http://ro.ecu.edu.au/cgi/viewcontent.cgi?article=1101&context=ism)
2. "Software, Vendors and Reputation: an analysis of the dilemma in creating secure software", (Wright, 2010c) Paper presented at the *Intrust 2010*, Beijing, China.
3. "A Quantitative Analysis into the Economics of Correcting Software Bugs", (Wright & Zia, 2011c), Paper presented at *Computational Intelligence in Security for Information Systems* (CISIS 2011), Spain. An **ERA B** Conference.
4. "Rationally opting for the insecure alternative: Negative externalities and the selection of security controls", (Wright & Zia, 2011d) Paper presented at *Computational Intelligence in Security for Information Systems*, Spain. **ERA B** Conference.

Chapter 4

5. "Of Black Swans, Platypi and Bunyips. The outlier and normal incident in risk management", (Wright, 2010b). Published in the SANS reading room and presented in ISACA Oceania CACS (CACS 2011).
6. "A comparative study of attacks against Corporate IIS and Apache Web Servers", published in the SANS reading room (Wright, 2011a).
7. "A preamble into aligning Systems engineering and Information security risk", (Wright, 2011d),

http://www.sans.org/reading_room/whitepapers/webservers/comparative-study-attacks-corporate-iis-apache-web-servers_33734 .

8. "Who pays for a security violation? An assessment into the cost of lax security, negligence and risk, a glance into the looking glass". (Wright, 2011) Paper presented at the 2011 International Conference on Business Intelligence and Financial Engineering (ICBIFE 2011), Hong Kong. Extended from a poster Paper presented at the ACISP 2011, Melb. Aust. **ERA B** Conference.

Chapter 5

9. "The not so Mythical IDS Man-Month: Or Brooks and the rule of information security", (Wright, 2010a), Paper presented at the ISSRE, San Jose, CA, USA. **ERA A** Conference.
10. "Using Checklists to make better best", (Wright & Zia, 2011f) Presented at the 9th Australian Information Security Management Conference (SecAU 2011). **ERA B** Conference.
11. "Rewarding IT staff in a changing environment to promote secure practice." Presented at SANS Brisbane 2012.
12. "HRM, it's not just hiring for compliance" Presented at SANS Brisbane 2012.
13. "Human Resources Management: Hiring the Right People to Remain Secure" Presented at SANS Brisbane 2012.

Chapter 6

14. "Criminal Specialization as a corollary of Rational Choice", (Wright, 2011b) Paper presented at the ICBIFE, HK, China.
15. "Territorial behaviour and the economics of botnets", (Wright, 2012) Presented at the 10th Australian Information Security Management Conference (SecAU 2012). **ERA B** Conference.

Conclusion

16. Wright C (2012), “Effective Strategies to Manage People and Processes to Leverage Current Investment in Security”, ACS Journal (Presented in an extended form as the conclusion) - Safe and sound. Information Age September/October 2012, Pp. 68–69.

IV Research Objectives

The primary research aims expressed through this thesis involve the demonstration of the ability to measure and quantify risk within information security. It was hypothesised that through the integration of measurements around the human factors, the system and design factors and the software aspects of risk that it would be possible to quantitatively measure risk and create game theoretic frameworks enabling organisations to determine how they could most efficiently allocate funds against the control of that risk.

This research is far from the and creation of complex multivariate survival models that can be deployed within multiple organisations and in differing scenarios. The creation of such a product and solution would be far outside the scope of any such individual research. This research however has demonstrated conclusively that the creation of such models is possible. In demonstrating methodologies for the creation of risk experiments and measurements related to system security and reliability engineering and in demonstrating the overall effectiveness of many of these controls we have shown that it is possible to create a risk management system that can be used to probabilistically determine the level of risk faced by an organisation.

Chapter 1 Introduction

1.1. Introduction: human, design or software.

Absolute security does not exist, nor can it be achieved. Even ostensibly secure systems are vulnerable, as all systems possess a level of insecurity—an attacker with sufficient resources can always bypass controls. The goal is to ensure that the economic constraints placed upon the attacker exceed the perceived benefits to the attacker to mitigate the risk. The difficulty, however, lies in quantifying these risks to information systems.

Relative computer security can be measured using six factors (Aycock, 2006):

1. What is the importance of the information or resource being protected?
2. What is the potential impact, if the security is breached?
3. Who is the attacker likely to be?
4. What are the skills and resources available to an attacker?
5. What constraints are imposed by legitimate usage?
6. What resources are available to implement security?

The result is that security is a relative risk measure related to organisational economics at the micro level and the economics of national security at the macro level. This works to frame security in terms

of one's neighbour. The question is not, "am I secure?" (Wright & Zia, 2011e), but rather, "am I more secure than my neighbour?" In this thesis, it is shown that attackers are rational economic actors. As such, they will maximise their gains and minimise the risk to themselves in seeking gains from the systems they attempt to exploit.

For decades, information security practitioners have engaged in qualitatively derived risk practices due to the lack of scientifically valid quantitative risk modelling methodologies. This has led to a system failing "not ultimately for technical reasons but because incentives are wrong". Here "security failure is caused at least as often by misaligned incentives as by technical design mistakes" (Anderson et al., 2007). This misallocation of valuable resources and the corresponding decrease in the levels of protection for many systems not only make systems less secure in the present, but limit future expenditure on security controls (Wright, 2011e).

This dissertation assesses the individual security costs for a set of common risks and threat agents, and proposes a series of models for the assessment of security risk in economic terms. This assessment is vital in determining the cost-benefit of costly security controls in systems in general and software in particular. Using a combination of modern scientific approaches and advanced data mining techniques, this research is aimed at creating a game-theoretic quantitative model for information systems risk that incorporates both contract theory and the methods developed within behavioural economics.

It has been noted that "from an adversary's perspectives, this security strength, in combination with the personal risk of the attack to the adversary's reputation, safety, or freedom, are the metrics of interest when evaluating the security of a prospective target" (Schechter, 2004).

Strength of a security control can be measured as the amount of time needed to compromise a system without being caught and the subsequent length of time that the attacker can maintain control of a compromised system.

This research will demonstrate that the sanctioned lack of sharing limits the ability of markets to operate and hence to create an optimal price mechanism. Consequently, black markets arise where criminal and other undesirable actors openly collaborate to exploit vulnerabilities and compromised systems. As these groups, can be shown to act rationally in the pursuit of profit, one effective long-term risk-minimisation strategy is to reduce the incentives by making cyber-crime less profitable.

System vulnerabilities are derived from either human, design (or architectural) or software risks. This thesis will demonstrate that all risks to an information system can be expressed as a function of these three factors.

1.2. Significance of the Study

This study is significant for several reasons. The current inability to adequately assess the extent of criminal involvement in the sphere of information security poses grave security risks and hinders the development of countermeasures. Accurate, efficacious risk measurement has broad economic applications. In addition, a comprehension of information security risk is paramount to organisational management. Organisations adopt security practices for three primary purposes:

1) to restrict and govern cost through a lessening of those risks associated with the dissemination of sensitive business information and/or personal information,
2) to align business objectives with information security standards for compliance and other purposes, and
3) to safeguard the sustainability of the organisation.

Choi et al. (2005) investigated a model with negative network security externalities to examine the optimal software vulnerability disclosure decision of a software vendor. They reported that companies frequently announce vulnerabilities at non-socially optimal times. The optimal moment for disclosure of software security vulnerabilities was further analysed by Arora et al. (2004). The authors ascertained that vendors constantly decide to release a patch later than would be socially optimal. They do not, however, extend this research into modelling the optimal patching strategies or quantifying the risk associated with these patch processes.

Jaisingh and Li (2005) investigated the function of commitment in optimal social policy for disclosure of vulnerabilities. In this case, the vendor determines the patch-release time after a vulnerability is discovered, leading to a finding that the time lag between the decisions of a social planner and the software vendor is significant only where the attacker can accumulate an advantage from prior vulnerabilities over time. They did not extend their research into modelling the economics of that cost. In this thesis, I extend this research to estimate the cost efficiency of criminal botnets.

Cavusoglu et al. (2006) demonstrated that an organisation's patch cycle may not be synchronised with the software vendor's patch release cycle. They used this result to demonstrate that cost sharing and liability

schemes may aid in coordinating the patch process. The role of externalities in a network environment was explored by August and Tunca (2006), who investigated the role of policies in maximising "the value generated by software and highlight that consumers' purchase (or usage) decisions play a fundamental role in our results, as does the vendor's profit maximization" yet again have not modelled these cost functions.

Nicastro (2005) described how patching may involve days of effort where malware can take only minutes or hours to spread through unpatched systems (e.g., Moore et al. (2003) and Shannon & Moore (2004)). I look at methods of modelling such types of risk and propose controls that can minimise these control failures.

Most importantly, I provide evidence to show that security controls are an investment and not a sacrifice. The need for security has a direct correlation to the expected returns from the project, the possible losses and the project's impact on the organisation.

Taken this way, security is a scarce resource that organisations compete to maintain. Like all scarce resources, competition leads us away from seeking an absolute measure of cost towards a relative measure of value.

Security investment can be defined as any investment by the organisation in an individual project or organisational function (including individual systems) that increases the survivability of that project or function, and hence increases the project or function's individual chances of success at the cost of the organisation's ability to invest in other projects or functions.

1.3. Problem Statement—Defining Security and Risk

This research is the first step in a continuing effort to model and quantify information security risk and costs. The primary goal of this research is to create a set of models that can be used to model information security risk. This is a wide domain and it is necessary to limit the scope of this research presented in this thesis. The models presented here can be used to calculate many crime-ware statistics including:

- The expected and optimal economic size of botnets,
- the value of zero day and other exploits, and
- the profitability and defensibility of compromised systems.

To accomplish the research goals, it is necessary to recognise that information security is a risk function (Longley & Kwok, 1994). Paying for too much security can be more damaging in economic terms than not buying enough. This leads to decisions about where the optimal expenditure on damage prevention should lie. This research may be used to inform those responsible for important security-related decisions better. The conclusions will be presented using an empirical study of software hazard rates and audit failures along with the question of how to enforce liability in a global economy.

The research is intended to address some of the economic issues that arise due to an inability to assign risk correctly and failure to measure risk, and to look at the misalignment of information systems audit and the compliance regime. It addresses the externalities that restrict the development of secure software and how the failure of the end user to apply controls makes it less probable that a software vendor will enforce stricter programming controls, with concomitant failures in the audit and measurement processes. This includes a look at the misalignment of audit

to security. This misalignment results from the drawing of funds from security in order to provide compliance with little true economic gain (Wright, 2011e). Two significant circumstances include where costs are coupled with the incidence of definite events in this case, the total cost accrues as a step function, and selected cases may shift between varieties of states over time. Here cost can be seen to accrue at a constant rate determined by the state at a point in time. Here I consider estimation of the mean cumulative cost over a period of significance using techniques derived from the marginal structures of the cost process coupled with intensity-based models. I discuss robustness as it applies to adaptive censoring (Cheng et al., 2007) within the perspective of the multi-state methods. Data from a survival study of Honeynet systems conducted as a part of this research has been used to demonstrate these techniques.

Table 1 Definitions of levels of prevention

Level	Definition
Primary Prevention	Primary prevention strategies intend to avoid the compromise of a system. Most population-based activities that are designed to stop attacks against large classes and groups or clusters of systems are primary preventive measures.
Secondary Prevention	Secondary prevention strategies are used to diagnose and address the damage from an existing incident in its early stages before it results in significant loss. Such a strategy would incorporate early containment and mitigation of malware.
Tertiary Prevention	These actions reduce the negative impact of established compromises through a restoration of the system's function and through a reduction in complications associated with the incident.
Quaternary Prevention	This term describes the set of maintenance endeavours that mitigate or circumvent the negative consequences of unnecessary or excessive interventions with respect to an information system.

Costs or benefits (profit, for example) include payments to consulting and recovery firms, incident handling, and the loss of business during periods of attack and compromise, as well as the cumulative quality measures that can be associated with partially recovered systems (such as

a system that has been recovered from a malware incident without having been restored from reformat and install).

These measures can be used to evaluate solutions for the treatment of compromised systems that remain in production. Costs and benefits can be multivariate and can accumulate for various reasons. Costs could, for instance, be incurred through prophylactic solutions (including anti-virus or host-based IDS), therapeutic solutions (including patching and updates), and other preventative solutions.

Table 2 Mapping levels of prevention

Prevention levels			**Security Responder's side**	
			Malware, attack code or other compromise	
			absent	present
System User's side	Vulnerabilities and possible exploits	absent	**Primary prevention** (Vulnerabilities absent compromise absent)	**Secondary prevention** (Vulnerabilities absent compromise present)
		present	**Quaternary prevention** (Vulnerabilities present compromise absent)	**Tertiary prevention** (Vulnerabilities present compromise present)

Alternatively, costs can be accrued through the loss of system uptime involved in maintaining preventative solutions as well as through the requirements for redundancy design to avoid suffering an economic loss when maintaining systems. Patching systems can require reboots and other downtime where commercial activities must be taken offline and the economic returns from the system are diminished. In this context, curative solutions can be perceived to be fixes following an incident, and palliative solutions can be said to cover any preventative measures (including IPS, system hardening and awareness training).

A distressed system is one where some level of functionality has been compromised through an ongoing or partially rectified incident. In this case, the system will operate with a limited level of functionality (such as a system that has been recovered but not rebuilt following a compromise or an attack). This system could have been repaired but have damaged and unrecovered drivers such that it operates inefficiently or requires periodic reboots.

Table 3 Definitions of tiers of prevention

Tier	Definition
Universal Prevention	Addresses the entire population (national, state, local community, company) and has the purpose of preventing or delaying an entire class of incidents or events. All systems and people (through awareness and education), without screening or censoring, are provided with the tools, information and skills required to prevent a particular event or even incident.
Selective Prevention	Focuses on groups or classes of systems whose risk of developing particular classes of problems or where exposure to a vulnerability is elevated. The subcategories may be distinguished by physiognomies such as installed software, patching levels or use.
Indicated Prevention	Incorporates a screening process where selected systems are validated further or isolated based on indications of compromise or signature matches for other problem behaviours. Identifiers may include systems in open and un-firewalled environments, users who must access untrusted web sites, and systems where users can upload extensive amounts of information.

The term "costs" is frequently used to denote cost or other accumulative measures such as utility, profit, or system reliability, and **C(t)** denotes a cumulative (univariate) cost for an individual system over the period **(0, t)**. It is usual to include a random variable **T** that signifies the time interval associated with the cumulative process. Thus, the objects of importance are represented as **T** and $\{C(t), 0 \le t \le T\}$. Simple methods for the analysis of cumulative cost (D. Y. Lin, Feuer, Etzioni, & Wax, 1997) have focused either directly on these costs, or in limited instances solely on the total existence cost, **C(T).** A more effective and enlightening approach involves calculating not only the costs themselves

but also the causal event processes that produce these costs. In this way, cumulative utility could then be employed to characterise a systems quality or utility measure, so that **C(T)** can be thought of as a "quality-adjusted" existence. In this way, one can bring systems risk studies into alignment with many of the techniques used in epidemiology and related disciplines.

A modelling of the event processes that generate costs integrates:

- increased understanding and knowledge of the event,
- the facility to manage inspection designs that entail:
    - truncation,
    - intermittent observation, or
    - censoring,
- techniques that allow for cost prediction; and an ability to detach the underlying event process from costs. This is important in the case where costs may be subjective, or subject to conflicting interpretation.

Using this set of techniques, it is possible to both create and evaluate models on which analysis of cumulative costs can be founded, and then discuss the efficiency and robustness properties coupled with each technique. The following section introduces some general notations and illustrates two frameworks to construct cumulative processes. The first framework is made using an assumption where, for each system in the risk study, a cumulative cost process and a corresponding time when the process terminates is incorporated. This framework can be defined for both cost and quality based models.

1.3.1 Developing Models

In developing a study of the cost for an incident or other event, T_i is used to correspond to the extent of the system downtime. Where a compromise leads to total and unrecoverable system failure (for instance, the system needs to be rebuilt from format), T_i would correspond to the time when the system was compromised or otherwise taken offline (such an example would include the CMOS attacks[i] in the past).

As a study, will likely not include compromised systems (and especially hosts that have been completely disabled), the value T_i can be expected to be frequently right-censored at some censoring time τ_i. Here, it is expected that the true cost process will remain unobserved for $t > \tau_i$. Researchers in epidemiology and medical survival studies (Bang & Tsiatis, 2000; Gosh & Lin, 2000; D. Y. Lin, et al., 1997; Strawderman, 2000; Zhao & Tsiatis, 1997) have focused on the nonparametric estimation of the distribution of "total existence cost", $C_i = C(T_i)$. This is commonly simplified (Ghosh & Lin 2000) to be represented as $E(C_i)$ by these authors.

Generally, in deployment and when used to represent live systems, T_i cannot be independent of the cost process. In this case, if $C_i(t) = \{C_i(u), 0 \le u < t\}$, represents the cost history to time instance t, then the extinction time hazard function,

$$\lim_{\Delta t \to 0} \frac{P(T_i < t + \Delta t \mid T_i \ge t, C_i(t))}{\Delta t}, \qquad \text{Equation (1)}$$

can be shown to depend on $\overline{C_i}(t)$. This relationship necessitates that were the censoring time τ_i and $\left(T_i, \overline{C_i}(T_i)\right)$ independent, the censoring value $C_i^* = C(\tau_i)$ and the total existence cost $C_i = C(T_i)$ are not independent.

The second framework is constructed to model the multi-state cost process. Supposing that for any time t a system occupies one of K existence states, then it can be assumed that all systems are started in an uncompromised state[ii] 1 at $t=0$ and that states $1,...,(K-1)$ are ephemeral with state K being an absorbing state.

If $Y(t)$ corresponds to the state occupied by a system at time t, it is reasonable to suppose that a cost function $V[Y(t),t]$ exists where for short periods between $(t,(t+dt))$ the incremental cost function can be evaluated.

The overall cumulative cost until time t may then be represented as:

$$C(t) = \int_0^t V[Y(u),u]\,du \qquad \text{Equation (2)}$$

The progression concludes through extinction upon entry to state K at time T such that $V[K,u]=0 \;\; \forall \;\; (u>0)$. Consequently, one can create risk models if one categorises and classifies the systems and attacks and divides these into classes.

1.3.2 Classification

A Random Forest (RF) algorithm is an ensemble of unpruned decision trees. They are commonly deployed where there are extremely large training datasets and an exceedingly large quantity of input variables. In security and risk, the dimensionality can run into the thousands of input variables. An RF model generally comprises up to hundreds of individual decision trees. Conventional dissimilarity measures that work for simple risk data may not be optimal in modelling security risk. The use of dissimilarity measures that are based on the intuition of multivariate normal distributions (clusters have elliptical shapes) are generally found not to be optimised in modelling risk. This makes it desirable to have a dissimilarity that is invariant under monotonic transformations of the expressions derived from the risk metrics.

The RF dissimilarity focuses on the most dependent markers, whereas the Euclidean distance focuses on the most varying marker. The more important a system is per RF, the more important it is for survival prediction. This allows the security risk practitioner to select systems based on quantitative measures rather than perception.

The primary benefit to risk modelling is that Random Forests tend to be very stable in model building. Their relative insensitivity to the noise that breaks down single decision-tree induction models makes them compare favourably to boosting approaches while they are generally more robust against the effects of noise in the training dataset. This makes them a favourable alternative to nonlinear classifiers like artificial neural nets and support vector machines.

As the performance is frequently reliant on the individual dataset, it is a good practice to compare several approaches.

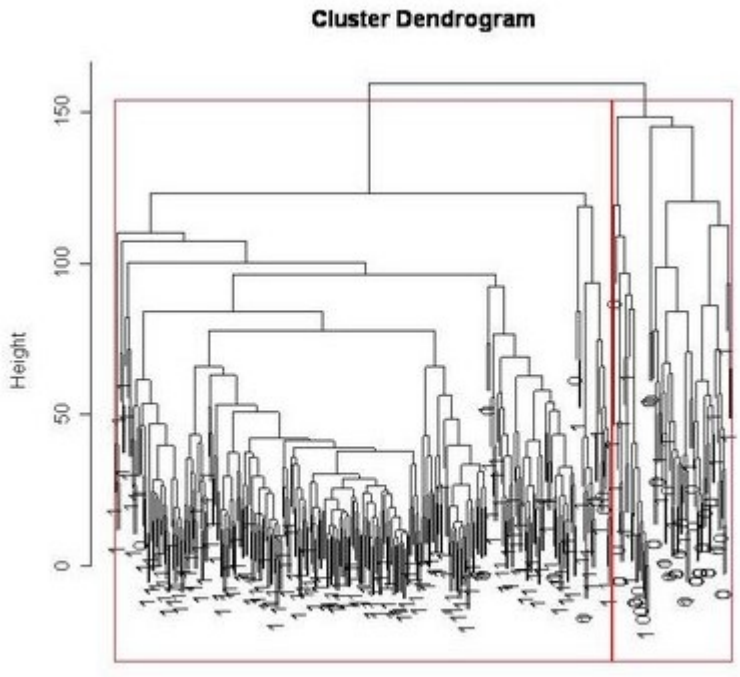

Figure 1. A cluster dendogram using a random forest algorithm.

Each decision tree in the forest is constructed using a random subset of the training dataset using the techniques of bagging (replacement). Several entities will thus be included more than once in the sample, and others will be left out. This generally lies in the two-thirds to one-third ratios for inclusion/exclusion.

In the construction of each decision-tree model, an individual random subset of the training dataset uses a random subset of the presented variables to decide where to partition the dataset at each node. No pruning is performed as all decision trees are assembled to their maximum magnitude. The process of building each decision tree to its maximal depth results in a less biased model.

The entirety of the decision-tree models taken together form the forest (Fig. 1). In this, the forest characterises the final ensemble model. Each decision tree in this model effectively casts a vote, with the majority outcome being classified as the outcome. In the case of regression models, the value over the ensemble of regression trees is averaged to produce the assessment.

A random forest model is effective for building Security Risk models due to several reasons:

1. The amount of pre-processing that needs to be performed on the data is minimal at most,
2. The data does not need to be normalised and the approach is resilient to outliers,
3. Variable selection is generally not necessary if numerous input variables are present prior to model building,
4. All the individual decision trees are in effect independent models. When taken with the multiple levels of randomness that exists within Random Forests, these models tend not to overfit to the training dataset.

1.3.3 The Challenge

Like all aspects of the economy, information security can be represented through an economic risk function. Excess spending on security can be more damaging in economic terms than not purchasing sufficient controls. The difficulty is always in determining where optimal expenditure on damage prevention should lie and in adjusting this efficiently in a dynamic market.

1.3.4 Purpose of Study

This thesis seeks to address some of the economic issues that arise due to an inability to correctly assign risk in the information security domain. It examines the externalities that restrict the development of secure software and how the failure of the end user to apply controls makes it less probable that a software vendor will enforce stricter programming controls. I combine hazard models with SIR (Susceptible-Infected-Removed) epidemic modelling to provide a means of calculating the optimal information systems audit strategy and economic models for cyber-crime, and around malware. Treating audit as a sequential test allows for the introduction of censoring techniques that enable the estimation of benefits from divergent audit strategies. This process is demonstrated to be effective at gauging the economic benefits of these strategies in the selection of an optimal audit process designed to maximise the detection of compromised or malware-infected hosts. A selection of models and methods that are common in many other areas of systems engineering, but which are only just starting to be used in the determination of information systems risk, are presented alongside the idea of using neural networks of hazard data to reliably model and train risk systems and hence allow organisations to better determine the correct risk strategy.

From all of this, it is evident that there is a need to develop robust methods of modelling risk in information systems. In its execution, this study represents the preliminary work in information security research for developing a systematic technique for collecting, analysing, and interpreting data about information security risk from an economic perspective.

1.4. The Structure of the Thesis

The thesis is organised as follows. I start by introducing the topic, covering the state-of-the-art, introduces information security risk and shows why this type of research is so important. This section includes a review of the information security literature as well as introducing the terminology used within this thesis.

I introduce the terms, the problems and the nature of the Internet as a series of interconnected systems that can be addressed as a mathematical network, and propose several methods to measure risk in such systems.

The second chapter examines the structure and significance of the study and provides a literature review, and Chapter 3 examines modelling software risk and looks at such risk through a quantitative and economic lens.

Chapter 4 details research on systems and threats, and Chapter 5 addresses in greater depth the human resources aspect of the problem, and considers how checklists and incentives (Lui et al., 2011) can bolster organisational security practices. Chapter 6 presents research that quantitatively models cyber-crime. Chapter 7 is the thesis conclusion.

1.5. Methodology

The purpose of this research has been to model a variety of approaches to risk and Information Systems Security. In this exercise, several studies have been formulated to demonstrate the interrelationships between the various aspects of risk as they relate to Information Systems Security.

These are separated out into individual components. The primary aspects of information security concern three factors:

- The risk associated with Information Systems and design,
- that which is derived from the human aspects of information technology, and
- risk and vulnerability that has derived from software.

Each of the experiments was conducted as a separate exercise and was published in a focused manner related to the individual results. The primary method of research centred on the audit of existing systems and implementations. The methodologies utilise for these audits have been widely used within industry and follow the standards set by SANs[iii] and ISACA[iv]. The systems audit methodology was derived using the principles followed within COBIT 4.0.

This thesis presents several studies based on audit results as well as supporting experiments. The studies help to measure the level of software risk in existing organisations and analysed the effectiveness of audit and controls. A limited experiment was conducted by extending general audit practice to analyse the results of prior audits against existing ones.

A simple experiment measuring the effectiveness of a single control was conducted to determine conclusively whether individual controls could make a quantitative measurable difference to the security of the system through a reduction of risk. To this end, honeypot systems were configured using standard industry practice with many of the systems running as controls with the Windows firewall disabled as was common at the time from many systems and then analysing these against the alternative of replicated systems with the sole change being the addition

of turning on the firewall. This type of experiment allowed us not only to measure whether a control was effective but to model the survival rates for the systems.

These experiments demonstrate that probability functions can be mapped for individual controls and systems. Through the introduction of risk functions associated with individual controls coupled with the impact of co-linear effects across competing controls it would thus be possible to create systems that measure and analyse failure.

The systems used within the software audits are common practice based on SANS, ISACA and OWASP[v]. When a virtualised machine or honeypot was created, the study would follow the methodology detailed by Bednarski & Branson (2004).

An appendix has been included at the end of the thesis detailing the processes involved in auditing and the general flow for information collection utilised within the experiments and studies.

1.6. Conclusion

The problem with quantifying information systems risk comes from addressing the various components of information security in isolation. This research explores this problem and demonstrates that the human, system and software aspects of information risk and security are not independent and need to be modelled in conjunction with each other. In incorporating these aspects, organisations will be able to manage risk at a lower cost.

No organisation is the same and the models created will be dynamical and vary in time and with the value of the information being defended.

This does not lead to a state of not being able to manage risk, but rather an understanding that simple compliance models are not adequate and that organisations need to aim to ensure that risk processes need to focus on security and not compliance goals to be successful.

Chapter 2 Background and Literature Review

2.1. Introduction

Information security is an economic function. Security can be modelled as a maintenance or insurance cost as a relative function but never in absolute terms. As such, security can be modelled as a cost function that leads to the prevention of loss, but not one that can create gains in utility (or profit).

Risks can originate from factors related to humans, design, or software. Criminals exploit such vulnerabilities in a way that can be considered rational (even if it is socially deviant), since such actions are in the criminals' self-interest (namely, the pursuit of profit). I will demonstrate that one effective means of countering this is to reduce the incentives for engaging in cyber-crime by minimising its profitability.

Molloy, Cheng & Rohatgi (2008) have taken a similar approach to that presented in this thesis. The authors assert that "*the field of information security should be viewed as a problem of risk management, where risk is roughly defined as the expected values of damages and treated as a countable and finite resource; the damages are the possible outcomes of*

security decisions and actions". This economic approach leads to the implementation of market based controls. Unfortunately, there exists a wide-ranging scepticism of market based solutions (Kayser & Budinich, 2015) despite the continuing successes and returns that are achieved using a market based solution. In this, it is critical to remember that market based controls are not deterministic, but return results probabilistically.

The consequence is that it is essential to understand compromise. Security controls are not absolute and losses will occur. In this, the risk practitioner needs to accept that the investment in a security control may not be justified, not in that it is not effective, but that alternative investments are more economically attractive and hence would lead to improved long term returns.

2.2. Information security as an economic risk function

Information security is a risk function (Anderson, 2001; Longley & Kwok, 1994) and hence can be addressed in economic relationships. Paying for too much security can be more damaging in economic terms than not buying enough. Just as with insurance, where paying more for the value of the good than can be recovered is a loss, so is the expense on security controls that provide no additional economic benefits. Where a dollar spent on additional security provides less than a dollar in additional benefits, a loss has occurred. This thesis presents ongoing work toward measuring the effectiveness of audit and assessment as an information security control. This is achieved by the creation of methods that can be used to quantify the costs and benefits of these controls (Davis & Holt, 1993).

The trend towards the application of security control measures that are employed to express compliance with legislation or regulations, rather than to explicitly detect or prevent breaches, results in a misallocation of funds. The inclusion of additional security controls leads to diminishing returns provided by each additional control. The costs of implementing each additional control remain static. That is, when valuable resources are expended on that return which is less than the expected value of the control, a loss occurs. This thesis reveals several major misconceptions among businesses about what security means and how that compliance is pursued to the detriment of security. It is easier to measure compliance than it is to measure security, and spending money to demonstrate compliance does not in itself provide security.

What information security and risk practitioners all need to be asking is who should be responsible for the security failures that are affecting the economy and society, and how can this be maximised to minimise negative externalities? Externality, or the quantitative and qualitative effects on parties that are affected by, but not directly involved in a transaction, is likewise seldom quantified (Cheng et al., 2007), but is an integral component of any risk strategy. The costs (negative) or benefits (positive) that apply to third parties are an often-overlooked feature of economics and risk calculations. For instance, network externality (a positive effect that can be related to Metcalfe's law [Van Hove, 2014], that the value of a network = 2x the network's number of users) attributes positive costs to most organisations with little associated costs to itself.

In these calculations, the time-to-market and first-mover advantages are critical components of the overall economic function, with security playing both positive and negative roles at all stages of the process. These externalities also have negative effects, such as those in the software industry where the "Ship it Tuesday and get it right by Version 3"

(Anderson, 2001) approach has become commonplace. This archetype has also moved into other negative externalities (such as malware with the advance of botnets). In the past, malware was perceived to be little more than a nuisance by many infected users whose hosts would send spam emails to others and grow the infection. As the user was not directly affected himself, there was often a low incentive for the user to react. This would lead to the installation of known malware (such as Bonzo-Buddy[vi] and Wack-a-mole) being actively installed by many users. Interdependent risk issues also apply and must be incorporated into any valid quantitative risk model for information systems risk. The issue of "free riding" (where another party gains from the economic expenditure of another party) is also a critical feature of this form of model.

Choi et al. (2005) investigated a model with negative network security externalities to examine the optimal software vulnerability disclosure decision of a software vendor. They reported that companies frequently announce vulnerabilities at non-socially optimal times. The optimal moment for disclosure of software security vulnerabilities was analysed by Arora et al. (2005), who ascertained that vendors constantly decide to release a patch later than would be socially optimal. Jaisingh and Li (2005) investigated the function of commitment in optimal social policy for disclosure of vulnerabilities. In this case, the vendor determines the patch-release time after a vulnerability is discovered, leading to a finding that the time lag between the decisions of a social planner and the software vendor is significant only where the attacker can accumulate an advantage from prior vulnerabilities over time.

The role of externalities in a network environment was explored by August and Tunca (2006). The authors investigated the role of policies in maximising "the value generated by software and highlight that consumers' purchase (or usage) decisions play a fundamental role in our

results, as does the vendor's profit maximization". Nicastro (2005) described how patching may involve days of effort while malware can take only minutes or hours to spread through unpatched systems. The result is unpatched systems that can be used to spread attacks widely and as attack platforms. Criminal groups can then use these compromised systems to further attack those groups that have expended the effort to patch.

Brito et al. (1991) produced one of the earliest papers to regard individual incentives and negative externalities concerning the spread of infectious diseases. They noted that forcing vaccination decreases social welfare. In these models, rational agents behave in a manner consistent with the social objective. Geoffard and Philipson (1996) emphasised the dissimilarity amid economic models and mathematical epidemiological models, a disparity that can produce economic loss and a misallocation of scarce resources. Economic models of infections can be applied to the spread and modelling of malware with regards to the role of rational agents when weighing protection costs and negative externalities (Hofmeyr et al. 2011).

This research looks especially at the misalignment of audit to security, which results from the divergence of funds from security to provide compliance with little true economic benefit. If information security is treated in the same manner economically as insurance and other risk-hedging methodologies, one sees that the expense applied to unnecessary controls results in a net loss.

2.2.1 Absolute and relative

All systems exhibit a level of insecurity. The goal is to ensure that the economic constraints placed upon the attacker exceed the perceived benefits to the attacker (Cavusoglu et al., 2006). This consequentially

leads to a measure of security in terms of one's neighbour. The question is not, "am I secure?", but rather, "am I more secure than my neighbour?" At the organisational level, a firm maximises its security over information by ensuing that it is a less profitable target to cyber-crime than its competitors. Similarly, national security is best achieved in educating and managing information and creating an environment that leads to the economically sustainable development of secure practices.

This can be assessed in many ways, as any other system is your neighbour on the Internet when viewed from the perspective of a worm. Conversely, targeted attacks have a purpose. Neighbours (here being referred to as adjacent systems or infrastructure) may be other government systems, critical infrastructure, and a class of companies or an industry sector. In each instance, security is achieved in relative terms. As such, to create an optimally secured system, it is necessary to address some of the economic issues that arise due to an inability to correctly assign risk (JASON, 2004, p. 50).

2.3. Markets and their role in creating secure practices

Arora and Telang (2005) asserted that a market-based mechanism for software vulnerabilities will necessarily underperform a CERT-type mechanism. The market that they used was a game theoretic pricing game (Nisan, 2007). In the model, the players in the market do not report their prices. These players use a model where information is simultaneously distributed to the client of the player and the vendor. The CERT model was touted as being optimal. It relies on waiting until a patch is publicly released and only then releasing the patch to the public. This ignores many externalities and assumes the only control is a patch in place of

other alternative compensating controls. Fundamentally, Arora and Telang's (2005) proposed model ignores the pre-existence of criminal marketplaces and other externalities that can skew the results away from a desired condition outside the laboratory.

Criminal groups have an incentive to maximise the time that vulnerabilities remain unknown (DSD, 2012; Williams, Dunlevy & Shimeall), as this extends the time that they have to exploit these bugs. Penetration testers have similar incentives as the trade secret of an unknown zero-day vulnerability can provide them with a competitive advantage. This also touches on reputational motives and pricing for both parties. The examined "market" model (Arora and Telang, 2005) creates incentives to leak information without proper safeguards and creates black markets specialising in the sale of exploits for system and software vulnerabilities (Radianti & Gonzalez, 2006). As criminal groups and selected security vendors (such as penetration testers and IDS vendors) have an incentive to gain information secretly, they likewise have an incentive to pay more for unknown vulnerabilities in a closed market. This means that a seller to one of these parties has a reputational incentive to earn more through not releasing information, as the individual's reputation will be based on their ability to maintain secrecy.

This misaligned incentive creates a sub-optimal market. As a consequence, the market reported (Arora, Krishnan, Telang, & Yang, 2005; Kannan & Telang, 2004) was sub-optimal to a CERT due to its inefficiency (Schalb, 2007) and not that markets are in themselves less effective. The skewed incentivisation structure recommended in the paper was the source of the inefficiency and not the market itself. This simply highlights the need to allow efficient markets (JASON, 2004, p. 46) to develop rather than seeking to create these through design.

The efficient market hypothesis (which asserts that financial markets are "informationally efficient") as touted by Paul Samuelson (1972) has come under heavy critical review by behavioural economists and others using quantitative studies (Basu, 1977; Nicholson, 1968; Rosenberg et al., 1985). In this theory, an investor cannot consistently achieve returns in excess of average market returns on a risk-adjusted basis, given the information accessible at the interval the investment is completed. The alternative, the inefficiency hypothesis, asserts that the market prices of common quantities and goods deviate from an ideal and "correct" market rate that would, in the case of a security for instance, be valued at the proper discounted rate of its future cash flow. Inefficient markets are also stated to operate inefficiently where volumes of trades are insufficient to adequately commoditise the product.

The inefficient market hypothesis when applied to information security leads to the contention that market forces drive asset prices towards a level that is both unsustainable and lies either above or below the proper price. Arora et al. (2005) found support for their argument in instances of market falls and bubbles. The existence of these is argued as evidence justifying intervention into the market process (under the premise that an overseeing body or government can hold more information in a diverse marketplace and act more rationally than the combined set of players in the market).

Due to such "inefficiencies", market prices for security quantities (such as vulnerability markets and even controls) can become or remain either overpriced or under-priced and lead to arbitrage profits or losses. In Chapter 3, it is demonstrated how an open market leads to a low opportunity for such arbitrage opportunities when information is made available and the quantities can be openly traded. In this view, the

"inefficiency" arises not from an inherent inefficiency, but from restrictive interventionist controls.

The other argument posed arises as a consequence of information asymmetry. Arora et al. (2004) asserted that software vendors have an informational advantage over other parties. The vendor does have access to source code (which is also available for Linux and other open source providers), but it can be proved that this does not provide the levels of information asymmetry that are asserted and that information asymmetry is not inherently biased in favour of either the software vendor nor of the user of a product. Software vendors have a reputational input to their value (Cavusoglu et al., 2006). Telang and Wattal (2004) did note that the market value of a software vendor exerts influence through reputational costs and those vulnerabilities correlate significantly with a decrease in the company's traded price, a view supported by others (Cavusoglu et al., 2006). Telang & Wattal (2004) state:

> "Vulnerability disclosure adversely and significantly affects the stock performance of a software vendor. We show that, on average, a software vendor loses around 0.63% of market value on the day of the vulnerability announcement. This translates to a dollar amount of $0.86 billion loss in market value. We also show that markets do not penalize a vendor any more if the vulnerability is discovered by a third party than by the vendor itself."

These results demonstrate that a vendor has an incentive to minimise the vulnerabilities found in their products; otherwise, it adversely affects market capitalisation against share prices. This justification offers strong evidence that a vendor does not have an incentive to hide information (as third party vulnerability researchers cause an equal loss in capitalisation). It should be expected that any vulnerability known by the vendor will be

uncovered. If the vendor fixes the flaw before release, the cost is minimised and at the limit approaches the cost of testing (that is, a zero-incremental cost to that which would be expressed later). If the vendor discovers a vulnerability in the software they produce, the result is a "strongly dominated" motive to fix the bug. Hence, any remaining bugs are those that have not been uncovered by the vendor and which are less economical to find (through an increase in testing). It can thus be demonstrated that the vendor knows no more than the user at the point of software release as to the state of bugs in a product.

Testing is far less expensive earlier in the development cycle (Tassey, 2002; Telang, & Wattal, 2005). Figure 2 below displays the expected utility of testing as the development progresses through an SDLC. Early in the process, the software developer has the greatest returns in testing and bug finding. As the development progresses, the returns are reduced as the process required and the costs associated with finding and correcting software vulnerabilities increase.

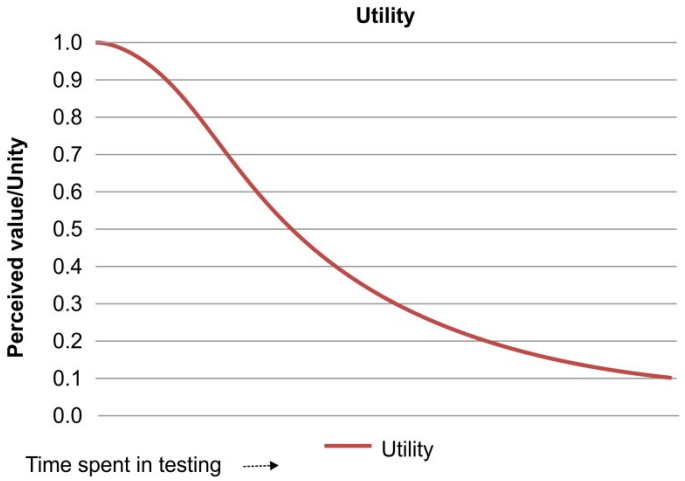

Figure 2. Decreasing utility of testing as the SDLC progresses.

The utility is the lowest when the software has been shipped to the user. At this point, fixing flaws is an expensive process for both the user

and the vendor. Figure 3 plots[vii] the total utility returned in expending time in testing. This leaves the optimal solution for the vendor based on the discovery of as many as possible as early in the development process as is feasible as a bug discovered early in the process can cost as much as 10 times less (Brooks, 1995) than one discovered later (Tassey, 2002). It does not mean that all bugs or vulnerabilities will be found, as the cost of finding additional vulnerabilities quickly exceeds the returns.

The widespread use of open source software (such as Linux) and the limited differential in discovering bugs using source code reviews demonstrates that the vast majority of software flaws in commercial software is discovered early in the development process (it is highly unlikely that commercial software developers are less effective than open source developers and likely that the rate of errors is likely to be similar if not lower).

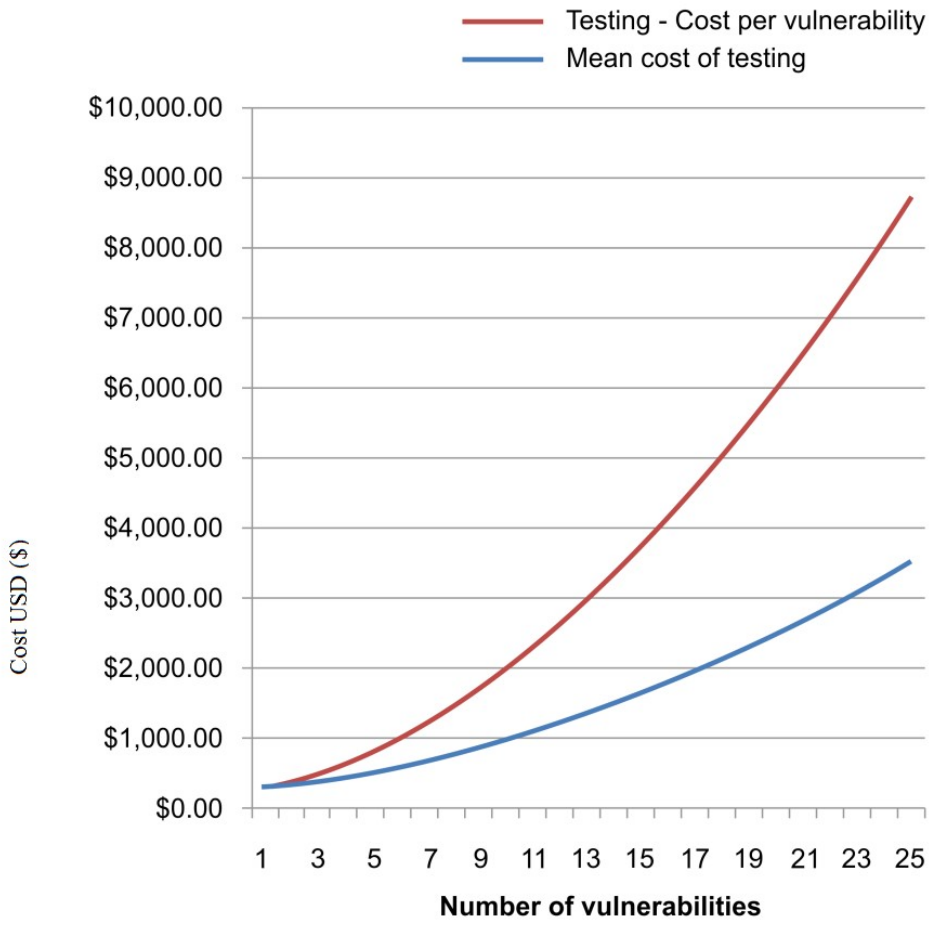

Figure 3. Each vulnerability costs more than the last to mitigate.

It has been estimated (Hein, 2011) that it takes 5,000 man hours of program use to discover the average software vulnerability, representing a significant expense for software development when it has to be conducted completely internally. An open market, on the other hand, distributes this expense and provides a source of information regarding the true cost of the vulnerability. This information is created as the true costs of developing patches and can be compared to alternatives where patching is costlier.

George Akerlof's (1970) model was designed for modelling quality uncertainty (Wright & Zia, 2011e) has been proposed as a game model

for the software industry (Schneier, 2007). This model is based on information asymmetry and the presumption that the vendor has more knowledge of the product than the user. This is demonstrated to be a flawed model in that the software vendor is incentivised to correct bugs as early in the process as is possible (the later a bug is discovered in the development process, the more it costs to fix). Hence, the vendor does not have a better sense of the expectations of flaws than a knowledgeable user. Further, the user knows how they plan to deploy the software; the vendor does not have this information and may have little insight into what other interactions may occur.

A vulnerability market[viii] provides information on discovery and vendor action (for instance Microsoft vs. Adobe vs. Linux etc.), allowing clients to better select software vendors and mitigate the "Market for Lemons" (Akerlof, 1970) that has been proposed (Schneier, 2007). It is demonstrated in this thesis that software vendors do not have more knowledge of bugs than users (or these would have been fixed prior to release) and hence do not have an advantage in information asymmetry when it comes to finding software flaws that may result through product interactions.

The market for lemons requires that the vendor knows the level of flaws better than the user. This may seem a common-sense outcome: the vendor has access to source code, wrote the program and ran the development process. However, this is a flawed view, as it is in the vendor's interest to mitigate vulnerabilities as early as possible. More importantly, the vendor is punished for bugs in both sales (Perrow, 1984; Weigelt & Camerer, 1988) and an indirect effect against the firm's share price (Telang & Wattal, 2004).

Yet other research has been formulated in order to create microeconomic models that are designed to model loss and allow firms to decide on the optimal level of expenditure that will maximise returns (Gordon & Loeb, 2002). Others have sought to set absolutes that can be directly measured against security risk (Hoo & Soo, 2000). These efforts, however, assume that security is an absolute, but as has been stated, security cannot be modelled as an absolute and must be constructed as a set of relative measures.

More recently, the move has been towards the creation of insurance (Fisk, 2002) and risk derivative markets (Blakley, 2002; Schneier, 2002) such as the industry already uses in the insurance markets (Molloy et al. 2008). These quantitative models of safety and reliability intend to provide a means to measure and hence quantify information systems risk (JASON, 2004) based on data-rich models. This leads to the use of complex hazard and survival modelling techniques that have been applied in fields such as epidemiology. Here, "big data" and advanced statistical data mining procedures are replacing the use of actuarial tables to gauge risk. In this approach, the forecasting of future event likelihoods cannot rely solely on historical data and large models of events across industry need to be mined. This underscores the need to share data to be able to create learning models (as discussed later in the thesis), as no actuarial table or risk model that can be created on historical data alone can correctly estimate the effect of selecting one security strategy over an alternative.

2.3.1 An approach using game theory

The introduction of game theory (Nisan, 2007) and behavioural economics has created a foundation for the rationalisation of information security processes that lead to improved allocation of economic

resources. The optimal distribution of economic resources across information system risk allocations can only lead to a combination of more secure systems for a lower overall cost. Incorporating the game theoretic multi-player decision problem allows one to assume that agents in the model are rational with well-defined preferences (such as in the case of criminal actors). One can then factor in the ability to strategically reason using their knowledge and belief of other players and to act per a combination of both economic "first thought" and deep strategic thinking. Solutions to these models can be found using a combination of the following game devices (Nisan, 2007):

- Equilibrium: evolutive (steady state) games,
- Heterogeneous sequential games,
- Rationalisability: deductive reasoning.

The models detail the existence of strictly dominating games where these exist in information security practices and propose methods to improve these models. Existing information security practices in existing organisations will be classified into the following game types throughout this thesis:

- Non-cooperative vs. cooperative game,
- Strategic vs. extensive game,
- Perfect vs. imperfect information.

2.4. Risk assignment and software contracts

Mitigation of liability is an optimal strategy for economic actors. Mitigation of damages is concerned with both the post-breach behaviours of the victim and the actions of the party to minimise the impact of a

breach. In a software parlays' form of risk contract, this would impose costs on the user of the software to adequately secure their systems. This is a trade-off. Before the breach (through software failures and vulnerabilities that can lead to a violation of a system's security), the user has an obligation to install and maintain the system in a secure state. The user is likely to have the software products of several vendors installed on a single system. Because of this, the interactions of the software selected and installed by the user span the range of multiple sources and no single software vendor can account for all possible combinations and interactions.

Any pre-breach behaviour of the vendor and user of software needs to incorporate the capability of the vendors to both minimise the liability attached to their own products, as well as the interactions of other products installed on a system. It is feasible to deploy one of several options that can minimise the effects of a breach due to a software problem prior to the discovery of software vulnerabilities. These include:

1. The software vendor implements protective controls (such as firewalls),
2. The user installs protective controls, or
3. The vendor provides accounting and tracking functions,
4. The vendor employs more people to test software for vulnerabilities, or
5. The software vendor adds additional controls.

Where more time is expended on the provision of software security by the vendor (hiring more testers, more time writing code etc.), the cost of the software needs to reflect this additional effort, hence the cost to the consumer increases. This cost is divisible in the case of a widely deployed operating system (such as Microsoft Windows) where it is easy

to distribute the incremental costs across additional users through the economics of scale. Smaller vendors (such as small tailored vendors for the hotel accounting market) do not have this distributional margin and the additional controls could result in a substantial increase in the cost of the program.

This is not to say that no liability does or should apply to the software vendor. Vendors face a reputational cost if they fail to maintain a satisfactory level of controls or do not respond to security vulnerabilities quickly enough or suffer too many problems. The accumulation of a large number of software vulnerabilities by a vendor has both a reputational cost (Telang & Wattal, 2004) to the vendor as well as a direct cost (Wright, 2010c) to the user (these costs include the time to install and the associated downtime and lost productivity). Consequently, a user can investigate the accumulation of software vulnerabilities and the associated difficulty of patching or otherwise mitigating flaws prior to a purchase. These costs are thereby assigned to new vendors even if they experience an exceptionally low rate of patching/vulnerabilities. The user can act with a knowledge of costs where it is efficient for them to do so. As users are rational in their purchasing actions, they will incorporate the costs of patching their systems into the purchase price.

The probability of a vulnerability occurring in a software product will never approach zero. Consequently, it follows that the testing process used by the vendor is expressible as a hazard model (Beach & Bonewell, 1993). In this, it is optimal for vendors to maximise their returns such that they balance the costs of software testing against their reputations (Weigelt & Camerer, 1988).

The provision of a market for vulnerabilities leads to a means of discovering the cost of finding vulnerabilities as an optimal function. In

this way, the software vendors maximise their testing through a market process. This will result in the vendors extending their own testing to the point where they cannot efficiently discover more bugs. The prices of the bugs that sell on market are definite and vendors must pay to either purchase these from the vulnerability researcher (who specialises in uncovering bugs) or increase their own testing. Vendors will continue to increase the amount of testing that they conduct until the cost of their testing exceeds the cost of purchasing the vulnerability. The potential pool of vulnerability researchers is larger than the potential pool of in-house testing roles. The pool of vulnerability researchers would also increase unto a point where the cost of training to become a vulnerability researcher matches the demand of the consumer to fund vulnerabilities.

This market also acts as an efficient transaction process for the assignment of costs not associated with negligence. The user still should maintain the optimal level of controls that are under their influence (installation, patching frequency and implementation of system level defences), whilst the vendor is persuaded to pay the optimal level of costs for testing and mitigation.

2.4.1 Quantifying Coding and Reliability

A small number of studies of coding processes and reliability have been conducted over the last few decades. The majority of these have been based either on studies of large systems (Connell, 2003; Mills, 1971) and mainframe-based operations (Connell, 2003; Mills, 1971) or have analysed software vendors (Levendel, 1990). In the few cases where coding practices within individual organisations have been quantitatively analysed, the organisations have been nearly always large telecommunications firms (Anderson, 2001; Carman et al., 1995; Kaaniche & Kanoun, 1996; Khoshgoftaar et al., 1996; Mills, 1971) or

have focused on SCADA and other critical system providers (Munson & Khoshgoftaar, 1992) or are non-quantitative approaches (Bacon et al., 2009; Sestoft, 2008).

However, these studies do not address the state of affairs within the clear majority of organisations. With far more small to medium businesses coupled with comparatively few large organisations with highly focused and dedicated large scale development teams (as can be found in any software vendor), an analysis of in-house practice is critical to both security and the economics of in-house coding as this is an area of software development that has been overlooked.

Some researchers have declared that it is less the technical failings of software as much as a misalignment of incentives (Lui et al., 2011) that produces insecure software (Anderson, 2001; Varian, 2000) and systems. They assert that the real cause of security breaches arises from the fact that parties suffering the majority of the consequences lack the economic power to set the controls. This is, that the real effect of a security breach can be more devastating for external parties than for the system owner, leading to negative externalities when the system owner does not adequately protect their system. For instance, the approach noted by Anderson (2001) takes software vulnerabilities as a direct and irreparable flaw. Anderson fails to note that other solutions to the problem exist. One such example would be the use of application firewalls and proxies to stop attacks against flawed software. Here, known attack signatures could be filtered instead of relying on patching the software vulnerability.

As the reach of the Internet continues to expand, internal coding functions are only likely to become more prevalent and hence more crucial to the security of the organisation particularly with respect to the organisation's information security.

2.5. Rationalising criminal behaviour

As organised criminal groups can be modelled using rational choice theory (Wright, 2011b), models will be presented whereby criminal groups can be shown to act as profit-seeking enterprises. This provides the ability to shift the economic returns away from this activity, which necessarily results in a lower quantum or decrease in the ratio of criminal activity. Criminal behaviour can be explained not only by social deviance, but by the pursuit of financial rewards (Adabinsky, 1983; Broadhurst & Grabosky, 2005; Neufeld, 2010). As a consequence, as criminal groups face few financial disincentives, a growing class of criminal specialists has resulted (Neufeld, 2010). The course of action to best minimise the online criminal threat can be linked to minimisation of economic returns from cyber-crime.

Rational choice theory can help explain the impetus behind cyber-crime. Rational choice centres on instrumental rationality (Adabinsky, 1983). This is the choice of the most efficient and effective means to achieve a set of ends. These ends include the acquisition of wealth or other scarce resources. The decision of an individual or criminal group to engage in socially detrimental criminal activity is not based on social deviance, but rather a perceived rational choice that such activities are most likely to provide the desired outcome. Cohen (1976) asserted that conduct is rational if it involves the choice of means to achieve an end.

Grabosky and Broadhurst (2005) argued that cyber-crime reflects any other profit-based criminal activity. In this way, it can be explained by three factors: motive, opportunity, and an absence of capable governance or guardianship. Any one of these can be shown to increase the costs

associated with criminal actions and hence reduce the profitability of cyber-crime. This reflects the state of mind of the "rational criminal".

Rational choice theory assumes that an agent can be modelled as *Homo economicus* (Archer & Tritter, 2000), an individualistic passive agent who is moved by the conditions experienced. These experiences and circumstances, coupled with the rational agent's assumptions, lead to a series of rational choices that are believed to be optional by the agent, but may be detrimental to society as a whole (Hechter & Kanazawa, 1997). "Rational man" is thus an "economic man". Instrumental rationality is used by the rational agent to gauge the ends and means that establish the actions used to plot one's course through life (Clarke & Cornish, 1985). Choice involves an active progression in which each agent evaluates (consciously or subconsciously) the benefits and costs, and then makes a continuing series of conscious decisions (Friedman, 1953). Each agent reflects on his or her current circumstances as evaluated against the attainment of his or her goals as a set of composite goods. That agent alone can establish whether the price can be afforded (Archer, 2000).

Rational choice theory is based on the assumption that an agent as *Homo economicus* holds particular sets of discrete, fixed, hierarchical predilections (Zey, 1998). The assumption is that the agent will select the action with the preferred outcome. This outcome (if achieved) optimises the difference between the costs and benefits associated with the action (Clarke & Cornish, 1985). Rationality is achieved in a series of actions that remain consistent with the agent's stable preference rankings in a manner that is designed to return the optimal relation between the goals and beliefs of the agent (Gordon & Ford, 2002). This ideally returns the largest set of composite goods (as determined by the rational agent) for the lowest cost.

Actions, including crime, can be “rational” for agents at the individual level, a dynamic that, when manifested on the group level, generates a variety of systemic social outcomes. At times (such as may result from cyber-terror and DDoS attacks), many of the ill effects are intended by agents. But more often, the result is an unintended consequence that may be either socially optimal or more commonly socially non-optimal (Neufeld, 2010). The undesired effects from the actions of socially focused agents (such as police and “white-hat” hackers taking vigilante action) can also frequently result in unintended sub-optimal social responses and other unintended consequences.

2.6. Making simple changes reduce risk

Several experiments have measured selected aspects of information security risk. One such experiment involved the introduction of checklists into the intrusion response process (Wright & Zia, 2011f) and another involved the measurement of patching processes across a set of standard organisations (Wright, 2011e).

2.6.1 Checklists and measuring performance

Considering the wide range of security papers calling for the use of checklists (Baskerville, 1993; Denzin & Lincoln, 1998; Dhillon & Backhouse, 2001; Martin, 1973), little research is available that quantifies the effects of using checklists to reduce the risk a system is exposed to. Checklists have evolved over time and proponents of lists (Elberzhager et al., 2009) have created checklists for about every conceivable situation.

Bishop and Frincke (2005) demonstrated that “security checklists cause both alarm (because they can replace thinking with conformance to

irrelevant guidelines) and delight (because they serve as aids in checking that security considerations were properly taken into account)", and use both analogy and anecdote to demonstrate that checklists can boost security testing.

Bellovin (2008) reminded us that a poorly structured checklist, "especially if followed slavishly or enforced without thought—can make matters worse." But little is also provided as to what effect a checklist can include. In this study, the individuals have been allowed to create their own checklist, both to minimise any potential aversion to using such a tool as well as to align this to the existing best practices of the individual and organisation.

2.7. Psychological bias and the human aspects of security

The inherent psychological biases that have developed in the information security profession have centred on the outlier effect (Wright, 2010b). This has led to a dangerously skewed perspective of reality and an increase in the economic costs of security.

The issue of moral hazard (Mougeot & Naegelen, 2009) where the principal or foremost agent is not informed of the agent's private information ex-post, is applied to insurance, monitoring, employee effort, and other aspects of information security. The model of the signalling game (where the agent tries to convey useful information to the principal) is investigated and is shown to be a means to signal ability (a secure system) whilst jointly avoiding the problem of free riding by other parties to the model.

The behavioural effect (Kahneman & Tversky, 1984) of loss aversion (defined as a propensity of information security professionals to minimise the impact of loss, even against risks that have expectation values of greater gain) is explored in association with concepts of social capital and cognitive biases such as the endowment effect (for instance, where an individual is "willing-to-reveal" at high price, "willing-to-protect" at low price). These issues are appraised against psychological propensities for both anchoring and adjustment and the status quo bias (the predisposition to resist changing an established behaviour, unless incentive is overwhelmingly compelling). Finally, the valence effect (as is associated with an individual's overestimation of the likelihood of favourable events being associated and impacting oneself) is used to model the association to the impact and causal relationship with respect of information security and the feedback effect from rational ignorance and "cold–hot empathy".

Yet other researchers have applied economic principles in a more politically socialised manner, seeking to set up a vulnerability pseudo-market for DoS, DDoS and other exploits in the form of a pollution or carbon credit scheme. This creation of a vulnerability credits scheme (Camp & Wolfram, 2000) is based on the notion that governments can arbitrarily and accurately decide what the optimal levels for vulnerabilities should be. In this approach, negative externalities perceived to be the cause of the issue and are left to government bodies to address rather than allowing the systems that cause the problems to bear the direct costs.

One shortcoming of this scheme arises from the interconnected state of the Internet; namely, that jurisdictional issues make such a scheme difficult to enforce. Thus, a small number of regulated entities are left to bear even more of the cost whilst computers in other jurisdictions are left

open as attack platforms. This creates a form of negative externality; exactly what the researchers sought to avoid.

2.8. Conclusion

In this thesis, information security is demonstrated to be a quantifiable risk that can be determined within the bounds of an algorithmic equation. The difficulty in this comes from the relationship between security and compliance. It is demonstrated in this research that compliance can have a detrimental outcome when tied to security.

The central management of dissimilar systems cannot be readily achieved. Rather, a distributed approach is required where each entity or organisation models the various aspects of their own environment based on the value of information being defended and the level of resources that are available to implement the tests and controls.

The research presented in this thesis both supports and extends the work of preceding researchers such as Cheng et al. (2007). In modelling aspects of risk, such as “the leakage of sensitive information by human users” (Cheng et al., 2007), the individual costs and functions of controls can be better understood. In this thesis, we extend this approach to demonstrate that all aspects of risk and security can be measured and monitored. It is further demonstrated that even where precise measurements cannot be achieved, the ability to retune and improve controls over time leads to reduced risk. As more information becomes available, the width of the confidence intervals for any risk equations an organisation develops will narrow, allowing for more accurate measurement of risk and hence more efficient control.

This thesis presents risk as an economic variable. In doing this, organisations can more effectively allocate scarce resources. Once the underlying proposition that all security incidents have a cost and that controls to minimise incidents require investment is understood, it becomes clear that security is not an absolute, but is an economic function designed to maximise the returns on investment. In this sense, when we understand that all investments can be deployed in a multitude of competing methods, that security controls are not simply a need, but are a means to an end. In this sense, security controls not only compete with other controls, but also with alternative uses of an investment.

Chapter 3 Modelling Software Risk

3.1. Introduction

Market models for software vulnerabilities have been disparaged in the past for their ineffectiveness. In this chapter, based on Wright & Zia (2010) and Wright (2010c), I argue that the market models proposed are flawed, not the concept of a market itself. A well-defined software risk derivative market would improve the information exchange for both the software user and vendor, removing the often-touted imperfect information state that is said to be the standard state of the software industry. In this way, users could have a rational means of accurately judging software risks and costs and vendors could optimally apply their time between delivering features and averting risk in a manner demanded by the end user. If the cost of an alternative control that can be added to a system is lower than the cost of improving the security of the software itself, then it is uneconomical to spend more time and hence money improving the security of the software.

In "A Quantitative Analysis into the Economics of Correcting Software Bugs" (Wright & Zia, 2011c), a quantitative study of in-house coding practices is used to demonstrate the notion that programming needs to move from "lines of code per day" as a productivity measure to one that takes debugging and documentation into account. This might be better conceptualised as "lines of clean, simple, correct, well-documented code per day", but with bugs propagating into the sixth iteration of

patches, a new paradigm needs to be developed. Finding flaws in software, whether these have a security-related cost or not, is an essential component of software development. When these bugs result in security vulnerabilities, the importance of testing becomes even more critical. Many studies have been conducted using the practices of large software vendors as a basis, but few studies have looked at in-house development practices. This section uses an empirical study of in-house software coding practices in Australian companies to both demonstrate that there is an economic limit to how far testing should proceed as well as noting the deficiencies in the existing approaches.

This chapter then develops a series of formulae that can be used to estimate the levels of software vulnerabilities, and concludes with a demonstration of how the game theoretic approach of a stag hunt can be used in modelling the inclusion of security controls in the software development process.

3.2. A legislative approach

Many information security professionals call for legislation and penalties against software vendors who have bugs in their software. This thesis examines the economic impact of several approaches to enforcing software security and demonstrates that a market-based approach is the most effective.

In most existing jurisdictions, the introduction of an exculpatory rule in contracts allows for the negation of a legal ruling (the default rule) where both parties have (through the contract) expressly consented to the contrary. Just as a car maker is not liable for the life of an engine exceeding 60,000 km unless there is an express warranty stating this, an

express exculpatory rule for software does not exist unless the vendor offers this in a warranty clause.[ix]

Legislation can make many superficial exculpatory clauses either invalid or redundant. Such exclusion would include negligence. In some cases, the exculpatory clause merely restates the existing perspective of the law. Where an exculpatory clause is permissible, the final assignment of liability does not depend on the preliminary distribution of liability. If the default rule is that a software vendor is liable for software vulnerabilities, but the vendor's profitability is greater where it does not suffer liability, then the vendor will print a label stating that is it not liable. In the case of software, this could be a click wrap agreement.

This chapter examines the economic impact of various policy decisions as a means of determining the optimal distribution of costs and liability when applied to information security and in particular when assigning costs in software engineering.

3.3. Risk assignment and Software Contracts

In economic terms, liability should be assigned to mitigate risk. The rule that creates the best incentives for both parties is the doctrine of avoidable consequences (marginal costs liability).

3.3.1 Software Derivative Markets

One possible solution to the limited and sub-optimal markets that currently exist would be the creation of hedge funds for software security. Sales in software security-based derivatives could be created on forward contracts. One such solution is the issuing of paired contracts (such as exist in short sales of stocks[x]). The first contract would be taken by a user

who would pay a fixed amount if the software had suffered from any unmitigated vulnerabilities on the (forward) date specified in the contract. The paired contract would cover the vendor. If the vendor creates software without flaws (or at least mitigates all easily determinable flaws prior to the inception of the contract), the contract pays them the same amount as the first contract.

This is in effect a "bet" that the software will perform effectively. If a bug is discovered, the user is paid a predetermined amount. This amount can be determined by the user to cover the expected costs of patching and any consequential damages (if so desired). This allows the user to select their own risk position by purchasing more or less risk as suits both the risk tolerance and the nature of the user's systems.

In an open market, such a derivative would indicate the consensus on the security of the software and the reputation of the vendor. Such an instrument would also allow software vendors and users to hedge the risks faced by undiscovered software vulnerabilities. These instruments would be in the interest of the software vendor's investors as the ability to manage risk in advance would allow for forward financial planning and limit the negative impact that vulnerability discovery has on the quoted prices of a vendor's capital.

3.3.1.1 Selective Survival

If the vendor creates the low feature version for $200 and the full featured secure version for $800, the user can choose the level of protection. Taking a survival model of Windows 98 and one for Windows XP (Campodonico, 1994), the client can calculate the expected difference from each product. If Windows 98 has an expected compromise rate (without other compensating controls) of once per 6.2 days and Windows XP has an expected rate of 52.5 days when the firewall is enabled,[xi] one

can calculate the cost per incident. If the user has a mean cost per incident of $320, one can see that the Windows 98 option has an expected annual cost to the organisation of $19,040 ($18,840 damage plus $200) whereas Windows XP has an expected cost of $3,825 ($3,025 damage plus $800).

Hence, most companies with a need to secure systems selected the more secure option and installed Windows XP over Windows 98. The initial costs did not reflect the overall costs to the organisation. Likewise, home users (who rarely calculate the costs of compromise) were more likely to install the cheaper option.

To extend the analogy, assume that a software vendor can create a perfect version of software for a mean cost of $20,000[xii] and the flawed version at the standard costs of under $1,000. Where the expected damages do not exceed $19,000 per copy of software, it remains in the user's interests to select the lower security option as the expected losses are lower. More generally, for an organisation with ***n*** users and an expected cost $C_S = nP_S$ (where P_S is the cost per user of the secure software product) against $C_I = nP_I$ (with P_I being the cost per user of the insecure software product), the cost of deploying either option is based on the expected losses for each option with a loss function defined as $D_S < D_I$. As such, if $(C_S - D_S) < (C_I - D_I)$ it is in the interest of the company to take the secure option. Where $(C_S - D_S) > (C_I - D_I)$, the economical solution is to install the less secure version of the software and self-insure against the loss.

As a means of maximising the allocations of risk and costs, the vendor could offer liability-free and full-liability versions of their product, with the latter being sold at a reduced price. The vendor could then control their risk using a hedging instrument. A market evaluation of the risk

being traded would provide better information than that which the vendor has alone.[xiii] The user could also purchase the same derivative as the vendor. The cost to the user to purchase the software plus risk insurance would be less than purchasing the "more secure" version from the vendor as the vendor would hedge the risk it faces and add a transactional cost as well as a return (at the firm's IRR).

Both parties are better off where the vendor produces the optimised version of their product and where the vendor does not assume liability for bugs.

Simply put, the vendor will provide a warranty (which acts as an insurance policy for the user) in cases where it can charge more for the provision of the warranty than it costs to provide the warranty. This cost could be either from a hedging instrument or through an investment in more testing. However, it must be noted that not increasing testing will make software invulnerable to all attacks and remove the ability for flaws to be discovered. As such, the cost of the hedged derivative will decrease, but will not go to zero.

The argument against this form of market is imperfect information. Many people argue that the software vendor has more knowledge than the user. This is untrue for several reasons. The vendor has no incentive to hide vulnerabilities as each vulnerability affects the share price of the vendor through a decrease in capital (Telang & Wattal, 2004). Next, the vendor would be competing on the same derivative markets as the users. The vendor's knowledge would be quickly and efficiently transferred through the market mechanism. The result is that a derivatives-based strategy does not allow for information asymmetry and would disprove entirely the assertion that software vendors operate a "market for lemons".

This demonstrates that the lack of liability does not require the software vendor to have the appropriate incentive to provide the optimal amount of testing and hence security (as well as to add the optimal levels of external controls) to the user. If more testing is cost effective in economically providing more secure software—that is, if the additional testing is cost effective—the software vendor will conduct this testing whether it is liable for security flaws or not.

The incentive to provide security is no different than the incentive to provide other characteristics and features. A vendor will add software extensions that can reduce the overall security of a system if the user values these to a greater extent than their costs.

Safety (and security in general) is a tie-in good. Amenities cost something. Vendors provide amenities if consumers (users) are willing to pay the cost of obtaining these. “Cars come with batteries and tires, but not always with stereo equipment or antitheft devices and rarely with driving gloves and sunglasses” (Donald, 2006).

What is demonstrated by those organisations with a defensive (and hence secure) coding regime is a lower variability in output. The mean coding time will remain similar, but the test time can be expected to be distributed differently for the organisation that codes with a defensive posture than that of the organisation that leaves too little time for testing. This increase in the standard deviation of produced results (or variability) increases the risk to the all parties (vendor and user).

3.3.1.2 Imperfect Information

At present, it can be demonstrated that neither side (the user or the vendor) is informed as to the true risks and costs of software. The vendor, for the most part, does not know the situation of the user and the user

may not be able to determine the risk from software. Worse, neither side is likely to know the risk posed from integrating multiple software products.

This asymmetry can be modelled. Take two hypothetical firms, SecureSoft and BuggySoft, each with 50% of the market for their software products, and give SecureSoft a mean time between failures[xiv] (MTBF) of 86 days and BuggySoft a MTBF of 17 days. In each case, the cost of a compromise is determined to have a mean of $400 (with $\sigma = 20$).

Next, assume that the average user (and purchaser of the software) cannot tell the difference, such that the user sees an MTBF of 52 days for either software vendor. This situation is known as 'adverse selection' in economics. If the users are liable and they self-insure, they will not be able to determine the relative level of vulnerabilities with respect to SecureSoft or BuggySoft. The cost of purchase will only be probabilistically determined (although, there are only small odds that an overlap event[xv] will occur) when the software vulnerability is uncovered.

If the software vendor were liable for more than one (1) vulnerability each 90 days, one would expect SecureSoft to breach around 40% of the time for an expected cost of $1,622 per annum. BuggySoft, on the other hand, would be expected to breach 100% of the time for an expected cost of $8,588 per annum. The difference in value of $6,966 is what BuggySoft would have to pay to its customers each year.

As such, SecureSoft could market itself as a securer option. If the user-base did not believe that BuggySoft and SecureSoft were different, SecureSoft could offer $6,965 (less than the calculated loss of $6,966) worth of liability insurance to its users whereas BuggySoft would not be able to match this price (all other aspects being equal). Thus, even an

argument for enforced liability insurance has a weakness. In the absence of liability legislation, the more secure software vendor would still seek to offer insurance to its users where it is cost effective to do so. By doing this, SecureSoft could add benefit to its users by insuring for loss to a level that is not cost effective for BuggySoft. This price differential would gradually win business for SecureSoft and increase its market share.

The consequence is that, although users see an equal distribution of problems initially, over time the information asymmetry decreases and the market starts to favour the most secure provider. The difficulty in these circumstances is not comparing which product is more secure, but is an issue of trading security against features. If BuggySoft added a management interface included with its software package that users valued at $8,000, it would be able to outsell SecureSoft, even with the insecurities and added risk.

This example elucidates the inherent issue with software security. Users value many aspects of a product and do so against the overall value that they assign to the product. Thus, an insecure product with saleable features will commonly sell for more than the secure version (even where users do not actually use the added features).

3.3.1.3 Optimal Derivatives Design Under Dynamic Risk Measures

The game theoretic approach to this can be modelled looking at the incentives (Lui et al., 2011) of the business and programming functions in the organisation. Programmers tend to be optimists (Brooks, 1995). As Brooks (1995) noted, “the first assumption that underlies the scheduling of systems programming is that all will go well”. Testing is rarely considered by the normal programmer, as this would imply failure.

However, the human inability to create perfection leads to the introduction of flaws at each stage of development.

In the model presented by Brooks (1995), as a program moves to a "Programming Product" and then to a "Programming Systems Product",[xvi] there are incremental additions that extend the estimates of the programming team. At each phase these can be expressed as a function of effort expressed in the lines of code, l_c. One can express the mean effort $\bar{x}_{l_c}$ required to code several lines and the variability, σ_{l_c} in achieving this outcome. This allows us to represent the coding effort as a representation of the stages of development:

$$F(x)=9H\left(\bar{x}_{l_c},\sigma_{l_c}\right)G\left(\bar{x}_{l_c},\sigma_{l_c}\right)+\varepsilon \qquad \text{Equation (3)}$$

In $F(x)=9H\left(\bar{x}_{l_c},\sigma_{l_c}\right)G\left(\bar{x}_{l_c},\sigma_{l_c}\right)+\varepsilon$ Equation (3), $H\left(\bar{x}_{l_c},\sigma_{l_c}\right)$ is the function of systematising (Brooks, 1995) the code. The expression $G\left(\bar{x}_{l_c},\sigma_{l_c}\right)$ is the productisation (Brooks, 1995) of the code.

The derivative would require the firm to publish their productivity figures;

Lines of code per programmer (and standard Deviation): l_c, $\bar{x}_{l_c}$ and σ_{l_c}

Bug-incidence figures: $\bar{b}_{l_c}$ and σ_b

- Estimating and project rules/methodology
- Software design methodology
- Secure Programmer measures:[xvii] $\bar{t}$ and σ_t

Many see this as proprietary data they would be loath to share, but as more actors in the market take up the use of such a derivative in

managing their own risk, the incentives for others to follow increase. One can also demonstrate that as $\bar{t} \to \max_t$ and $\sigma_t \to 0$ the volatility in coding σ_{l_c} and σ_b decreases with the number of reported bug incidents $\bar{b}_{l_c} \to 0$ as $\bar{t} \to \max_t$.

More metrics would be required based on the methodologies and coding practices used with different computer languages expected to exhibit different responses.[xviii]

As the skill of the programming team ($\bar{t} \to \max_t \& \sigma_t \to 0$) increases through group training, secure coding practices and in-line testing, the volatility in coding $\sigma_{l_c} \to 0.$. As the incremental returns on $\bar{t}$ diminish as $\bar{t} \to \max_t$ and the costs of training continue to grow linearly,[xix] it is apparent that the optimal position for any software vendor is to have a skilled team that maximises returns. At this point, additional training becomes economically unviable and does not create further returns for the organisation.[xx] It is also apparent that it is in the interest of the firm to minimise the variance in skill levels between programming staff (aim to have $\sigma_t \to 0$). This of course adds costs, as junior staff would be less productive in that they would have to work with more senior staff[xxi] to both cross train and to ensure that the junior employees maintain an acceptable level of production and testing.

This exercise of joining junior and senior staff would create a return function described by Brooks (1995) as a "Time versus number of workers—task with complex interrelationships". The output of the junior/senior staff unit would be lower than the senior staff level (initially) but would lower variability and increase the skill levels of the junior staff and hence over time $\sigma_t \to 0$. Due to staff turnover[xxii] and the increasing costs of maintaining more skilled programmers, this would

also have a negative feedback on the vendor's production costs. This would also need to be optimised rather than maximised if the vendor were to maximise returns.

3.3.1.4 No More Lemons

The fallacy of the mainstream analysis of deflation can be used to demonstrate the fallacy of aligning a market for lemons to computer software. In this game model, people refrain from spending when they anticipate falling prices. As people expect prices to fall, they believe that they will get a better price if they consume later (or that they will get more secure software for the same price). This drop-in consumption results in a consequential price fall, and the whole cycle repeats. The consequent argument is that prices will spiral uncontrollably downward.

Robert Murphy (2009) addressed this issue and demonstrated its fallacy:

> One could construct an analogous argument for the computer industry, in which the government passes regulation to slow down improvements in operating systems and processing speed. After all, how can computer manufacturers possibly remain viable if consumers are always waiting for a faster model to become available? … The solution to this paradox, of course, is that consumers do decide to bite the bullet and buy a computer, knowing full well that they would be able to buy the same performance for less money, if they were willing to wait… (There's no point in holding out for lower prices but never actually buying!) (pp. 68–9).

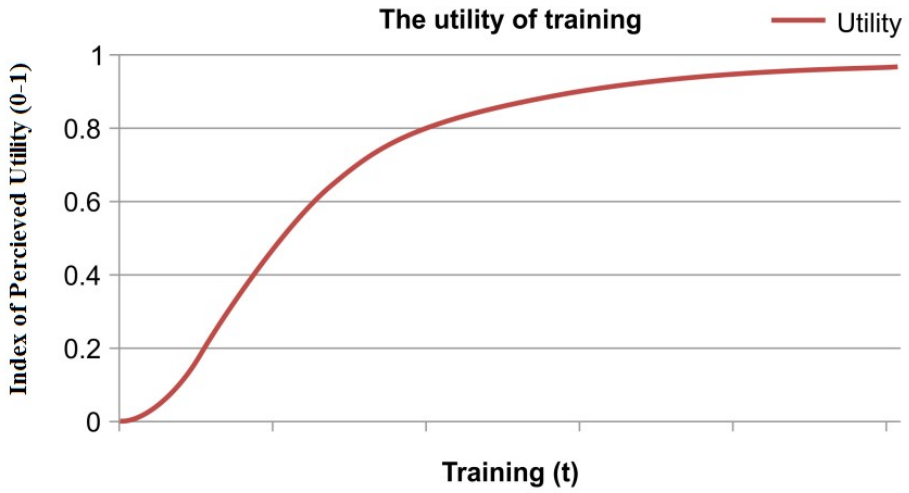

Figure 4. Training software developers in security adds utility.

Some users do wait for patches to be proven and others use "bleeding edge software" (and some never patch). The model is more rightly based on a distribution of users centred on an install time slightly after the release date.

This model can help us combat George Akerlof's lemons model[xxiii] of the used-car market that has been applied to the computer industry. Akerlof argued that asymmetric information would result in the owners or private vendors of good used cars being forced out of the market. Contrary to the equilibrium suggested in this model, good used cars sell. One can state that just because private vendors of good cars may not be able to obtain as high a price as they would like, it does not follow that they will refuse to sell at all.

Likewise, just as users do not know the state of security or how many bugs exist in software, software is still sold.

The "real balance effect": as prices fall, the value of money rises.

People’s demand to hold money can be satisfied with less money as price deflation occurs. This in part explains why people will eventually spend, even when they expect prices to continue to fall. The same process applies to software. Users see value in the features and processes they purchase software for. These are offset against the costs, which include both the monetary costs as well as the costs of maintenance (patching) and security (and other) flaws. Failure has a cost that is incorporated into the price of software to the user. The result is that users buy and install software even when they expect more bugs/vulnerabilities to be found.

3.4. A Quantitative Analysis into the Economics of Correcting Software Bugs

3.4.1 Introduction

Past studies into coding practice have primarily focused on software vendors. Many of these have been studies of in-house projects that are not incorporated into the practices and do not align well with in-house corporate code development. In the past, building software was the only option but, as the industry developed, the build vs. buy argument has swung back towards in-house development with the uptake of Internet connected systems. In general, this has been targeted towards specialised web databases and online systems with office systems and mainstream commercial applications becoming a "buy" decision.

As the web becomes increasingly indispensable in the corporate world, and as "cloud applications" become more widely accepted, in-house development expands. This thesis uses an empirical study of in-house software coding practices in Australian companies[xxiv] to both demonstrate that there is an economic limit to how far testing should proceed as well as noting the deficiencies in the existing approaches.

3.4.1.1 Related Work

Other studies of coding processes and reliability have been conducted over the last few decades. The majority of these has been based either on studies of large systems (Connell, 2003; Mills, 1971) and mainframe-based operations (Mills, 1971), or have analysed software vendors (Levendel, 1990). In the few cases where coding practices within

individual organisations have been quantitatively analysed, the organisations have been nearly always large telecommunications firms (Anderson, 2001; Carman et al., 1995; Kaaniche & Kanoun, 1996; Khoshgoftaar et al., 1996; Mills, 1971) or have focused on SCADA and other critical system providers (Munson & Khoshgoftaar, 1992) or are non-quantitative approaches (Bacon, et al., 2009; Sestoft, 2008).

3.4.2 Vulnerability Modelling

Vulnerability rates can be modelled extremely accurately for major products. Those with an extremely small user base can also be modelled, but the results will fluctuate due to large confidence intervals. What is most often overlooked is that the number of vulnerabilities or bugs in software is fixed at release. Once the software has been created, the number of bugs is a set value. What varies stochastically is the number of bugs discovered at any time.

This is also simple to model, the variance being based on the number of users (both benign and malicious) of the software. As this value tends to infinity (a large user base), the addition of any further users makes only a marginal variation in the function. Small user bases of course have large variations as more people pay attention (such as the release of software vulnerability).

This is a Cobb-Douglass function (Cobb & Douglas, 1928), with the number of users and the rate of decay as variables. For largely deployed software (such as Microsoft's Office suite or the Mozilla browser), the function of the number of vulnerabilities for a program given the size of the program can be approximated as a Poisson decay function.

3.4.2.1 Modelling the discovery of bugs/vulnerabilities in software

The discovery of software bugs can be mapped to the amount of time that has been used in both actively examining the product and the passive search for bugs (using the software).

The study found that a Cobb-Douglas function with $\alpha=1.6$ and (the numbers of bugs created in a code block) $F(x)= c \times TLOC + \varepsilon$ where c is a constant value with the function complete Cobb-Douglass function $G(x)^{\beta}$ remaining constant for a given number of users or installations and expresses the rate at which users report bugs. TLOC is the total lines of code. This equation increases to a set limit as the number of users increase. In the case of widely deployed software installations (such as Microsoft Word or Adobe Acrobat) and highly frequented Internet sites, this value tends towards $G(x)=1$.

3.4.2.2 Equations for Bug Discovery

For a static software system under uniform usage the rate of change, N, in the number of defects discovered is directly proportional to the number of defects in the system:

$$\frac{d}{dt}N(t) = \alpha N(t) \qquad \text{Equation (4)}$$

A static system is defined as one that experiences no new development, only defect repair. Likewise, uniform usage is based on the same number of runs/unit time. As the user base of the product tends to infinity, this becomes a better assumption.

By setting time T to be any reference epoch, then N satisfies

$$N(t) = N_i(T)e^{-\alpha(t-T)} \qquad \text{Equation (5)}$$

This means one can observe the accumulated number of defects at time t, A(t), where

$$A(t) = N(t)\left(1 - e^{-\alpha(t-T)}\right)$$

Equation (6)

With continuous development, an added function to model the ongoing addition of code is also required. Each instantaneous additional code segment (patch fix or feature) can be modelled in a similar manner.

We have created a reliability model that acts over testing time, *t*. We can measure the reliability of our software system using the failure intensity rate, $\lambda(t)$, which is associated with the measure of the total expected number of defect or bugs found in the software at time *t* using the value $\mu(t)$. This relationship between these two values is given by the expression, $\lambda(t) = \frac{d}{dt}\mu(t)$.

We define the total number of bugs or software vulnerabilities that have been detected at time *t* as being denoted by the value $N(t)$ under the assumption that on the discovery of a vulnerability the software vendor rectifies the existing flaw in the software. Here, we have not considered the introduction of further vulnerabilities in patches but are simply looking at the removal of existing vulnerabilities from the prior software set.

In this system, we can model vulnerability discovery using an exponential model based through an assumption that the rate of discovery is proportional to the number of defects present within the software. In such a model, the constant of proportionality is frequently represented using the value β_1 which acts as our constant of proportionality. In this

model, it is widely reported (Kuo & Yang (1996), Levendel (1990), Kay (1977)) that $-\frac{dN(t)}{dt} = \beta_1 N(t)$.

By substituting the value $\alpha = -\beta_1$, we come up with equation (4) which may be solved for $N(0)$, being the total number of defects that exist in the software at its creation time. Introducing this value, we get the relationship $N(t) = N(0)e^{-\beta_1 t}$, that we can represent as $N(t) = N(0)e^{\alpha(T-t)}$, leading to equation (5). Using these relationships, we can model the total number of expected faults that are detected at time *t* using the relationship $\mu(t) = N(0) - N(t)$ which then leads us to see that $u(t) = N(0) - N(0)e^{\alpha(0-t)}$ and hence that $u(t) = N(0)(1 - e^{-\alpha t})$. In moving to an epoch model of time from the time at the start of each software release, we obtain equations (5) to (12).

What one does not have is the decay rate and one needs to be able to calculate this. For software with a large user base that has been running for a sufficient epoch of time, this is simple.

This problem is the same as having a jar with an unknown but set number of red and white balls. If one has a selection of balls that have been drawn, one can estimate the ratio of red and white balls in the jar.

Likewise, if one has two jars with approximately the same number of balls in approximately the same ratio, and one adds balls from the second jar to the first periodically, that yields a mathematically complex and difficult problem, but one that has a solution.

This reflects the updating of existing software. In addition, with knowledge if the defect rates as bugs are patched (that is, the rate of

errors for each patch), one can calculate the expected numbers of bugs over the software lifecycle. In each case, the number of bugs from each iteration of patching added *34% ± 8%* more bugs than the last iteration.

$$A(t) = \sum_{i=0}^{k} A_i(t) = A_0(t) + A_1(t) + \ldots + A_k(t)$$
$$\approx A_0(t) + \beta A_0(t) + \beta^2 A_0(t) + \ldots + \beta^k A_0(t) \qquad \beta \le 1 \qquad \text{Equation (7)}$$

In the study, this would come to

$$A(t) = A_0(t) \sum_{i=0}^{6} (0.34)^i = 1.514 A_0(t) \qquad \text{Equation (8)}$$

So, over the life of the software, there are 1.51 times the original number of bugs that are introduced through patching.

A new software product means having prior information. One can calculate the defect rate per SLOC, the rate for other products from the team, the size of the software (in SLOC) etc. This information becomes the posterior distribution. This is where Bayesian calculations (Bayes, 1763) are used.

t = time

λ_B = (Mean) Number of Bugs / TLOC (Thousand Lines of Code)

L = SLOC (Source Lines of Code)

So, more generally, if a software release has L lines of code and the expected number of lines of code per defect is λ_B, then the a priori distribution of defects in the release is a Poisson P_β distribution where β is the ratio of new lines of code to average number of lines/bug (L/ λ_B)

$$P_\beta(n_{defects}) = \frac{\beta^n e^{-\beta}}{n!} \qquad \text{Equation (9)}$$

The conditional distribution for the number of defects in a software release given a defect discovery T units of time since the last discovery is

$$P_{\beta}(n\_defects) = \frac{\beta^n}{n!} e^{-\beta}$$ Equation (10)

Suppose the defect discovery (decay) constant is α and β is the a priori expected number of defects (code size/lines of code per defect). Observing defects at time intervals of T_1, T_2, ..., T_k, the conditional distribution of remaining defects is Poisson:

$$P_{(\beta e^{-\alpha(T_1+T_2+\ldots+T_k)})}\left(n_{defects}\right) = \frac{(\beta e^{-\alpha(T_1+T_2+\ldots+T_k)})^n}{n!} e^{-\beta e^{-\alpha(T_1+T_2+\ldots+T_k)}}$$ Equation (11)

This is the a priori expected number of defects scaled by the decay factor of the exponential discovery model.

As new releases to the software are made, the distribution of defects remains Poisson with the expected number of defects being the number remaining from the last release, γ plus those introduced, β, by the independent introduction of new functionality.

$$P_{(\beta+\gamma)}[n] = \frac{e^{-(\beta+\gamma)}(\beta+\gamma)^n}{n!}$$ Equation (12)

It is thus possible to observe the time that elapses since the last discovery of a vulnerability. This value is dependent upon the number of vulnerabilities in the system and the number of users of the software. The more vulnerabilities, the faster the discovery rate of flaws. Likewise, the more users of the software, the faster the existing vulnerabilities are found (through both formal and adverse discovery).

3.5. The Economics of Developing Security Embedded Software

This section presents a published paper that investigates the economics of security models and presents a hedge fund software risk derivative market approach to embed the security in software development. From the software security assurance perspective, the only practical approach to security assurance is software vulnerability scanning and testing. This approach of security assurance requires the software to be physically present, which requires already spending most of the budgeted time and cost in the development. If, after performing those tests, vulnerabilities are found, fixing those vulnerabilities would expend additional time and incur significant costs or may require the software to be scrapped and built from scratch again.

3.5.1 Related Work

Adams (1984) noted that a third of all software faults take more than 5,000 execution-years to manifest themselves. The secluded EAL6+ software sampled by Adams is not statistically significant over all software, but it does provide evidence of the costs. This also demonstrates why only two operating system vendors have ever completed formal verification. The "Secure Embedded L4 microkernel" by NICTA comprises 9,300 lines of code, of which 80% has been formally verified at a cost of 25 person years of work. The US$700 ploc costs for this exercise (the low estimate) demonstrates why formal verification is not a feasible solution for most software. This amount of time produces a great expense for software development when it must be completely conducted internally. An open market, on the other hand, distributes this expense in an optimal manner and provides a source of information as to the true cost of the vulnerability. This information is

created as the true costs of developing patches can be compared to alternatives where patching is costlier.

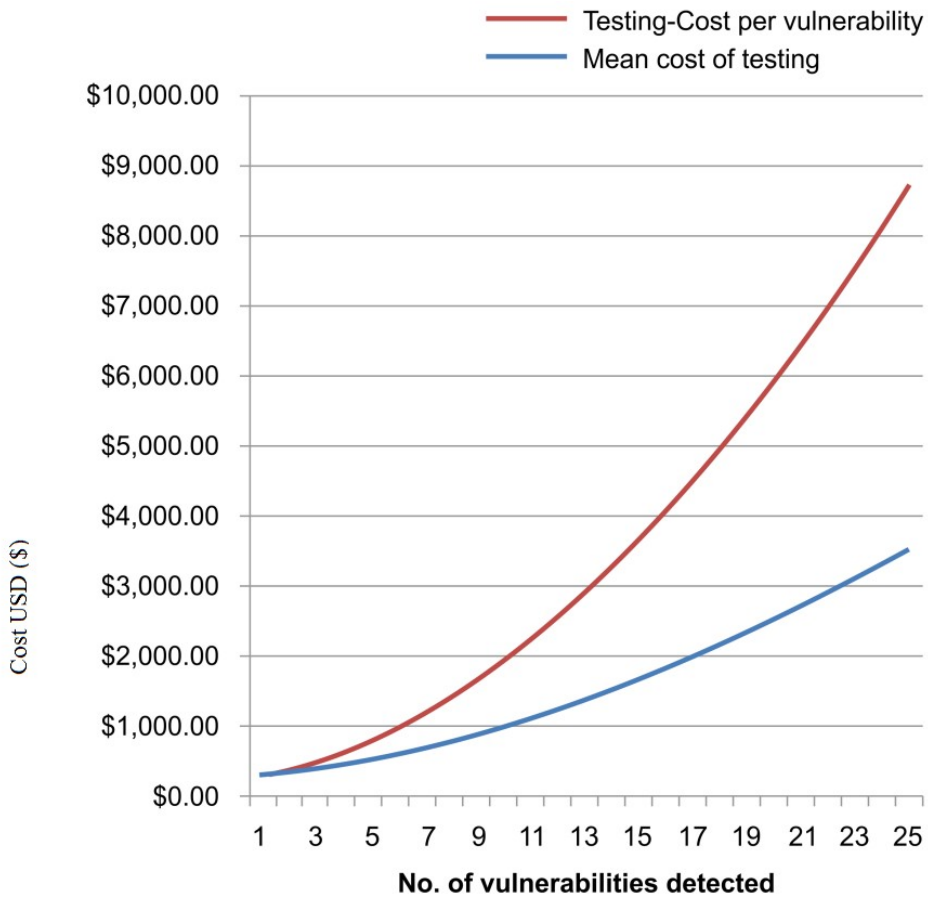

Figure 5. Each vulnerability costs more than the last to mitigate.

3.5.2 Selective Survival

It is also important that companies move from "lines of code per day" as a productivity measure to one that takes debugging and documentation into account. This could be something such as "lines of clean, simple, correct, well-documented code per day". This also has problems, but it contributes towards creating a measure that incorporates the true costs of coding. The primary issue is one of parsimony: the coder who can create a small, fast and effective code sample in 200 lines where another

programmer would require 2,000 may have created a more productive function. The smaller number of lines requires less maintenance and can be verified much more easily.

Such factors can be incorporated into a market-based model where small, clean code would be expected to be valued higher than a large code base with the same functionality, as the cost of maintaining the former is less than the latter.

It has been argued (Donald, 2006; Durtschi et al., 2002) that negligence rules are required to force software vendors to act optimally. However, the effects of reputation and the marginal cost effect from reduced sales are incentives in themselves for the vendor to act optimally. The supply of imperfect information to the vendor through incomplete and inadequate feedback channels can be resolved in part through the creation of a market for vulnerability research. The effects of reputation on the vendor and the assignment of risk through this process result in fewer negative externalities then occur because of a legislative approach. A market-based model, in contrast, allows both the vendor and the end user to evaluate the costs of security inherent in software and to evaluate alternative controls against competing products.

3.6. Rationally Opting for the Insecure Alternative: Negative Externalities and the Selection of Security Controls

3.6.1 Assessing Individual Security Costs

The most effective security solution is that which provides the best level (that which is optimised) for “the least cost”. Costs to the consumer are

minimised at the point where security costs exactly equal the expected loss that is associated with the risk function. In brief:

Higher security costs = higher costs to the consumer.

Higher expected loss from risk = higher costs to the consumer.

As expenditure on security is expected to lower the expected loss, the costs to the consumer are minimised where the additional expenditure of $1 on security reduces the expected risk-based loss by exactly $1.

Security is a cost function that is passed to the consumer if profitability is to be retained, or one that reduces profit directly where alternatives exist (this is where the product is elastic or consumers are willing to reduce their use if costs increase). The expected cost formula for the supply of these types of services against a loss function can be expressed by:

$$C_s = D(x, y) + x + y \qquad \text{Equation (13)}$$

Where the loss function $D(x, y)$ and the damage to x (the producer) and y (the consumer) are modelled arithmetically. As in all areas of economics, the marginal gains in D_x offset those of D_y.

In these calculations, $D_{xy}D_{xy} > D_{xx}D_{yy}$ represents the inference that the inputs are substitutes (Ni et al., 2010). As the producer spends more on security, the consumer spends less and vice versa. The exact composition of these values varies based on the nature of the product, with elastic supply being affected more than an inelastic supply.

The prevailing objective in security becomes the creation of a Cournot-Nash equilibrium (Kolstad & Mathiesen, 1991). This is an outcome where X_e and Y_e are together form a Cournot-Nash equilibrium

for a given value of Y_e; the x which maximises X's utility is X_e and given X_e that y which maximises Y's utility is y_e. This does not require that the equilibrium be Pareto optimal (Kurz & Hart, 1982).

At present, the cost functions directed towards many industries (such as banks in regulated countries, including Australia) are sufficient in that there is but a trivial increase in marginal demand for the consumer for an incremental increase in security expenditure. The producing company is likely to do little, and its actions have a minimal effect. For instance, Microsoft is unlikely to greatly improve the security of its operating system through minimising patches due to the increasing cost of finding additional bugs in its software. If it did so, Microsoft's profit would be diminished as consumers are generally unwilling to bear the cost increment that this would entail, making such an action economically infeasible. The incremental cost of finding additional bugs exceeds the total cost to all consumers of taking an alternative course of action such as installing HIDS (Host Intrusion Detection Software) and host firewalls.

The loss for the consumer is lessened to a lower extent than the loss of the producer. With fraud loss limits of $50 in countries such as Australia for online transactions, banks in these locations have an incentive to minimise the loss to the consumer. Perversely, this can incentivise the consumer against adequately securing their system. If the consumer expects to lose a maximum of L_{iy} (which is set at $50 for credit card transaction fraud in Australia) for any given incident *i* where the total expected damage is defined as:

$$D_y = \sum_{i=1}^{n} L_{iy} \qquad D_x = \sum_{i=1}^{n} L_{ix}$$

Equation (14),

the expected annual number of incidents per consumer *n* can be calculated as the total number of incidents that have occurred divided by the total number of consumers of a class (i.e. the total pool of credit card users).

$$E(n) = \frac{\#incidents}{\#consumers}$$ Equation (15)

Setting C_{Ty} as the total cost to the consumer of implementing controls, if the expected total loss to the consumer is $D_y < C_{Ty}$, it is doubtful that the consumer will pay for additional protection. For instance, if a high-end HIDS and anti-malware product costs $C_{Ty} = \$225,$ and the consumer experiences *n=4* incidents in a usual year, the expected damage $D_y = \sum_{i=1}^{n} L_{iy} = \$200.$ As $D_y < C_{Ty}$, it is not in the interest of the consumer to adequately protect their system. The user of a system that requires more security than the mean level of control provided by a vendor can implement increased security controls on their system, but this would either require that the consumer experience other measurable losses or that $D_y > C_{Ty}$ for this consumer.

Here it is evident that the anti-fraud efforts by banks and credit card companies create a negative incentive to consumers. The loss to the vendor L_{ix} currently averages $237 (Ben-Itzhak, 2009) for each lost set of credentials. The result is that it is in the interest of the financial company to provide the consumer with a compensating control. Holding the consumer liable if they had failed to use the enhanced controls over security would result in $D_y > C_{Ty}$, and hence an incentive for the consumer to protect their system.

Capital invested by the consumer in securing their system has a greater marginal effect than that of the producer in the case of an organisation such as Microsoft. A consumer can purchase HIDS and host firewall software for less than the cost that it would cost Microsoft to perfect their software through formal verification and hence remove more bugs.

The expected damage, $E(Damage)_i = P(x_{ai}).D_{Tot}$, is equal to the probability of a breach times the amount of damage suffered in a breach. This can be expressed as a function for each user or as a total cost function for all users, $E(Damage) = \sum_i (P(x_{ai}).D_{Tot})$. This reveals that the total amount of damage is a function of not only the producer, but also the consumer. The optimal solution is to find a point that minimises the total costs. This is the expected damage as a loss function plus the costs of damage prevention of a compromise of other loss. The damage can also be expressed as a function of both the producer and consumer (user) costs,

$$C_T = Cost_{Tot} = \sum_i [P(x_{ai})D(x_{ai})] + C_v + \sum_i [C_u(i)] \quad \text{Equation (16)}$$

The first order conditions are:

$$P'(x_{ai})D(x_{ai}) + 1 = 0 \quad \text{Equation (17)}$$

$$D'(x_{ai})P(x_{ai}) + 1 = 0 \quad \text{Equation (18)}$$

That is, the user should increase the expenditure on precaution (preventing a breach) until the last dollar spent on precaution by the user reduces the expected damage by $1. And the producer should increase the expenditure on reducing the possible damage in case of a breach until the last dollar spent on precaution by the producer reduces the expected damages by $1.

Clearly, the greater the likelihood of the user experiencing a breach, or the larger $P(x_{ai})$ is for the user, the greater the precaution that they should undertake. In the case of a producer who is a software vendor, they will (generally) sell their products to a wide range of users with varying likelihood that each will experience a breach. That is, the software vendor is acting with imperfect information.

The optimal amount of precaution is the solutions to Equations (13) and (16) and is denoted by the expressions C_{v}^{Ω}, $C_{u}^{\Omega}(i)$ and where the total costs for all users is optimised at $\sum_{i}\left[C_{u}^{\Omega}(i)\right]$.

The marginal utility expenditure of security means that the value of security decreases the more is added. There is a reason for this: in spending, more than the value of the organisation's capital, it is simple to see that the producer will not survive long. Hence, one only needs to reduce profitability for a producer to fail, not the capital.

The level of damages suffered by a user depends on both the pre-breach behaviour of the user and the vendor. The vendor is in a position where reputation impacts sales (demand) and hence the willingness to add layers of testing and additional controls (all of which increase the cost of the software). As the market for software varies in its elasticity (Stolpe, 2000) from the highly inelastic in small markets with few competitors (e.g. electricity markets) to highly elastic (e.g. operating systems), the user has the ability to best determine their needs. The user may select customised software with warranties designed to reduce the levels of breach that can occur. This comes with an increased cost.

Software vendors normally do not face strict liability for the damage associated with a breach due to a software vulnerability (Hahn & Layne-

Farrar, 2007; Scott, 2007). Although negligence rules for software vendors have been called for (Scott, 2007), this creates a sub-optimal outcome. The user can: (1) select different products with an expectation of increased security (Devanbu, 2000), (2) add external controls (through the introduction of external devices, create additional controls or use other software that enhances the ability of the primary product), and (3) increase monitoring for attacks that may be associated with the potentially vulnerable services such as by the use of IDS (Intrusion Detection System) (DShield, 2006–2010).

By limiting the scope of the user's responsibility, the user's incentive to protect their systems is also limited (Hahn & Layne-Farrar, 2006–2007). That is, the user does not have the requisite incentive to take the optimal level of precautions. Most breaches are not related to zero day attacks (DShield, 2006–2010). Where patches have been created for known vulnerabilities that could lead to a breach, users will act in a manner (rational behaviour) that they expect will minimise their costs (White & Dolin, 2006). Whether risk seeking or risk averse, the user aims to minimise the costs that they will experience. This leads to a wide range of behaviour, with risk-adverse users taking additional precautions while risk-neutral users can accept their risk by minimising their upfront costs, which may lead to greater loss later.

In any event, the software vendor as the cause of a breach is not liable for any consequential damages. This creates the appropriate incentives for the user to mitigate the risk. At the same time, the vendor has a reputational incentive to minimise the risk to their reputation. This was seen when the costs of bugs to the consumer from Microsoft (Hale, 2002) was deemed exceedingly high. The vendor response was to change their coding practices and to significantly reduce the number of vulnerabilities in their released code.

A better game model for the software industry is the "Stag Hunt," based on Jean Jacques Rousseau's postulations of a co-operation strategy between two hunters (as cited in Skyrms, 2004). These individuals can either jointly hunt a stag or individually hunt a rabbit. The largest payoff is assigned against the capture of a stag, which provides a larger return than the hare. The hunting of a stag is more demanding and requires mutual cooperation. If either player hunts a stag alone, the chance of success is negligible and sub-optimal. Hunting stags is most beneficial for society in that this activity creates the optimal returns. The problem with this game is that it requires a lot of trust among the players.

This game has two pure strategy equilibria in which both players prefer the lower risk equilibrium to the higher payoff equilibrium. The game is both Pareto optimal and Hicks optimal, but the sub-optimal and hence inefficient equilibrium poses a lower risk to either player. As the payoff variance over the other player's strategies is less than that of the optimal solution, it is more likely that this option will be selected. Another way of stating this is that the equilibrium is payoff-dominant while the other strategy is risk-dominant.

		Software User	
		Create Secure Software	Add Features
Software Vendor	Create Secure Software	10, 10 A, W	1, 7 B, X
	Add Features	7, 1 C, Y	5, 5 D, Z

Figure 6. Software markets as a "stag hunt".

The strategy between the software vendor and the software user is displayed in Figure 6, the numerical representations represent the payoff

figures for the specific case (the software market) and the generalised relations take the form:

$$A > C \geq D > B$$
$$W > X \geq Z > Y$$

Equation (19)

The outcomes are not definitive statements of what will be produced. In this game, the "stag" is a desire to "create secure software" and the "hare" the fall-back to adding more features. A desire is not a case of creating fewer bugs by itself, but rather a combination of adding controls and testing to software. Such an example would be the addition of the XP to Windows XP SP2 by Microsoft. Additional testing is effective to a point and more can be done than is occurring at present.

The payoffs for creating more secure software are great for both the vendor and the user, but the risk of a misaligned strategy leads to sub-optimal equilibria. What is needed is a signalling process. A signal will allow the players to align to the more optimal strategy. It is not only in the user's interest to have more secure software, but also is in the interest of the vendor. Patching is expensive and the vendor can reasonably charge more for secure software.

As the ratio between the payoff for stag hunting and the payoff for hare hunting is reduced, the incentives to move towards stag hunting decrease, making it less likely that software security will be made into a primary goal of either party. As such, where the introduction of special features and the "new killer app" occur more frequently, software security lags and it becomes more likely that a change from a stag hunting equilibrium to a hare hunting equilibrium will occur. It is hence less probable that an alteration of the player's strategy from hare to stag occurs.

Since neither player has an incentive to deviate, this probability distribution over the strategies is known as a correlated equilibrium of the game. Notably, the expected payoff for this equilibrium is 7(1/3) + 2(1/3) + 6(1/3) = 5, which is higher than the expected payoff of the mixed strategy Nash equilibrium.

3.6.2 Assessing Economic Value of Security

Being a relative function, not only does the profitability of an individual class (whether organisation, group or nation) factor into the calculation of security risk, but the relation to a class's neighbours also needs to be measured.

The cost function is in the criminal's favour without additional input from the consumer. There is no impetus for the bank to move to a more secure (and costlier) means of protecting consumers when the criminal can still gain access to the consumer's system. One part of the problem is the banks' own identity verification procedures. These enable criminals to pose as a representative of the bank and obtain confidential information from consumers, using social engineering and other less technical methods.

Whilst there are greater losses from consumer inaction than supplier inaction, the consumer's failure to secure their system and refrain from the use of systems at insecure locations exacerbates the risk of loss.

At all points of an assessment, one must also take the time value of money into account. The value of capital is not fixed and fluctuates with time. To evaluate costs, one must consider cost and the point at which the cost is expressed.

To compare any set of two or more alternatives, the financial characteristics of the alternatives must be compared on an equivalent basis. Two options are said to be equivalent when they have the same effect. Monetary values are termed as equivalent when they have the same exchange value. This can be defined as:

1. The comparative amount of each monetary sum,
2. The times that the occurrence of the sums can be aligned,
3. An interest rate can be used to compare differences in the time of payment.

The general equivalence function is defined as:

$$\text{PE, AE or FE} = f(F_i, i, n) \qquad \text{Equation (20)}$$

This equation holds for values of t between 0 and n. The equivalence equation uses:

$F_t =$ the rate of monetary flow at the end of time-period t,

$i =$ the rate of interest for the time-period,

$n =$ the number of discrete time periods.

The security and risk product lifecycle defines the function of the acquisition and utilisation phases. A system with a longer MTBF (Mean Time Between Failure) has a greater return on the initial investment. Similarly, larger upfront investments in security reduce the amount of capital available for investment. The financial present equivalent function [PE(i)] is defined as a value calculation that is related to the difference between the present equivalent capital value and the present equivalent costs for a given alternative at a given interest rate.

The present equivalent value at interest rate *i* over a total of *n* years is stated as:

$$PE(i) = F_0(^{P/F,i,0}) + F_1(^{P/F,i,1}) + \ldots + F_n(^{P/F,i,n})$$
$$= \sum_{t=0}^{n} F_t(^{P/F,i,t})$$

Equation (21)

Measures that take externalities into account act as a signalling instrument that reduces information asymmetry and improves the overall risk position of both the consumer and the vendor. The development of a software risk derivative mechanism would be beneficial to security (Jaziar, 2007) through the provision of a signalling process to security and risk.

In moving from security expenditure from a lower to higher value, the return on that expenditure increases to a maximum and then decreases. The optimal point is where security expenditure and expected returns result in positive growth. Before investing valuable resources into protecting the information assets, it is vital to address concerns such as the importance of information or the resource being protected, the potential impact if the security is breached, the skills and resources of the attacker and the controls available to implement the security. The value we must consider is not the capital, but rather expected return on capital. In any event, security expenditure fails where it costs more than it is expected to save.

3.7. Chapter Conclusion

As an investment, security can be more accurately costed using market models. Despite the criticism of such models, including vulnerability

markets, a more transparent pricing structure for the pricing of information security risk that openly displays the current cost of finding and fixing software vulnerabilities would be of great benefit. Not all users opt for security at the expense of features but where it can be demonstrated that the benefits derived from increased security outweigh either a cost saving or the perceived advantages of the additional features, security will increase.

Just as car dealers buff the exterior and detail the upholstery of a used car, neglecting the work that should be done on the engine, software vendors add features. Most users are unlikely to use even a small fraction of these features, yet they buy the product that offers more features over the more secure product with fewer features. The issue is that users buy the features over security. This is a less expensive option for the vendor to implement and provide.

The creation of a security and risk derivative should change this. The user would have an upfront estimate of the costs and this could be forced back to the software vendor. Where the derivative costs more than testing, the vendor would conduct more in-depth testing and reduce the levels of bugs. This would most likely lead to product differentiation (as occurred in the past with Windows 95/Windows NT). Those businesses willing to pay for security could receive it. Those wanting features would get what they asked for.

It is argued that software developers characteristically do not correct all the security vulnerabilities and that known ones remain in the product after release. Whether this is due to a lack of resources or other reasons, this is unlikely to be the norm and would likely be rectified by the market. The cost of vendors in share price (Arora & Telang, 2005) and reputational losses exceed the perceived gains from technical reasons

where the fix might break existing applications. The application is already broken in the instance of a security vulnerability.

Users could still run older versions of software and have few if any bugs. The issue is that they would also gain no new features. Users want features. They could also choose to use only secure software, but the costs of doing so far outweigh the benefits and do not provide a guarantee against the security of a system being compromised. As such, the enforced legislation of security standards against software vendors is detrimental. A better approach would be to allow an open market-based system (Molloy et al., 2008) where vendors can operate in reputational and derivative markets (Bacon, et al., 2009).

At the end of any analysis, security is a risk function, and what is most important is not the creation of perfectly security systems, but the correct allocation of scarce resources. Systems need to be created that allow the end user to determine their own acceptable level of risk based on good information.

To represent the effect of security expenditure in minimising bugs against investment over time and the result as expected returns (or profit), there are expenditure inflection points. Spending too much on security has a limiting function on profit; conversely, too little expenditure has a negative effect on profit as the cost of discovering bugs post-release increases. This is where risk analysis fulfils a much-needed purpose. The idea is to choose an optimal expenditure on security that limits the losses. Money should be spent on security until that last dollar returns at least a dollar in mitigated expected loss. Once the expenditure of a dollar returns less than a dollar, the incremental investment is wasted. Here, the software coder should optimise the testing process.

Modelling and understanding program risks are essential to minimise risk and create better code. It was clear from this study that organisational coding expresses a far higher rate of bugs per line of code than is expressed in specialised software companies. Insufficient testing is being conducted in many companies who have in-house coding teams, leading to higher costs and lower overall security. The goal for any coding team should be how many lines of good code are produced, not how many lines of code are written and then sent to be fixed.

When treated as an economic good, information security can be modelled as any other scarce resource. As information security requires skilled individuals such as trained developers, auditors and other security professionals, the time and cost of these individuals have a direct relationship to the amount of security an organisation can afford to deploy. Spending more on such individuals may increase the level of security and reduce the risk but at the same time will also reduce the amount of resources available for other investments. In this way, information security, like all security classes, should be an investment. Like all investments, there is an optimal balance, with too little expenditure leading to diminished returns and losses the same as when an excess is applied. In this way, either too little or too much leads to loss.

Many existing coding metrics, including “lines of code per day”, fail to consider the complete range of costs associated with a coding project. A different metric such as “lines of clean, simple, correct, well-documented code per day” would return more usable information to the organisation, allowing them to better determine the optimal expenditure point for information security. This point is a financial measure: spending a dollar more of security returns a dollar value, once this point exceeds any extra expenditure returns less than a dollar in benefits for each dollar of security expenditure.

Expenditure on security may be a cost function but it is not a sacrifice, economic or otherwise. The end goal of creating a secure system is to develop an infrastructure in which the optimal levels of profitability for any security project may be obtained. In this pursuit, the cost of security is measured as an investment where the returns on that investment should always be equal to or greater than the expected returns that the organisation associates with a successful project.

Chapter 4 Systems and Threats

4.1. Introduction

Chapter 4 starts with a comparison of attacks using known and zero-day vulnerabilities and exploits as published in "Of black swans, platypi and bunyips: The outlier and normal incident in risk management" (Wright, 2010b). One makes decisions in the absence of knowledge. One can state that black swans and bunyips do not exist. From time to time, this presumption proves erroneous when black swans are indeed found. However, for every black swan, there is a unicorn, dragon and bunyip that does not exist and of which one is confident will never be found.

The inherent psychological biases (Kahneman & Tversky, 1984) of loss aversion and people's tendency to strongly prefer avoiding losses to acquiring gains that have developed in the information security profession have centred on the outlier effect, dangerously skewing one's perspective of reality and increasing the economic costs of security. This paper demonstrates that producing resilient systems for known events also minimises the risk from black swans without the wasted effort of chasing myths.

This idea is extended in Wright (2011a). It has been suggested that an attacker[xxv] will specifically target the Windows operating system. This research has shown that, rather than this being the case, an attacker will in fact not target Microsoft Windows, but rather seek to avoid attacking Linux. Wright (2011a) has shown significant support for the assertion

that attackers shy away from Linux and not that they aim to attack Windows. This study was designed to collect data on attacks against corporate web servers. Unlike many of the other proceeding studies using honeypot systems (Pauna, 2014; McGrew, 2006; Choa et al., 2008), this experiment was designed to collect information on attacks against "secured" corporate systems. This data is used in support of the assertion that attackers target those systems that have the largest economic returns.

As is noted in Wright (2011e, 2011f) and Wright & Zia (2011a), information security is a risk function. Paying for too much security can be more damaging in economic terms than not buying enough. This next section addresses some of the economic issues arising from an inability to assign risk correctly. It looks at the externalities that restrict the development of secure software and how the failure of the end user to apply controls makes it less probable that a software vendor will enforce stricter programming controls. This chapter concludes with an exploration into the modelling of compliance and design risk.

4.2. Of Black Swans, Platypi[xxvi] and Bunyips:[xxvii] The outlier and normal incident in risk management.

The falsity of the black swan argument derives from a deductive statement that "every swan I have seen is white, so it must be true that all swans are white". The problem is that the swans that have been seen represent only a subset of the entire set. One cannot have seen all swans.

Likewise, the argument that not enough weight applies to zero-day vulnerabilities and that these are a major cause of system intrusions relies on the same reasoning. The assertion that more compromises occur because of zero-day vulnerabilities comes from a predisposition to

remember the occurrence of a zero-day attack more often than one remembers a more frequently occurring incident. While near eidetic recall comes from novel events, common events are more often forgotten (Roese & Olson, 2007). This leads to treating common events as if they matter less than they should.

Similarly, when Australia was finally explored, platypi and bunyips where reported along with black swans (Maish, 2013). At first, many refused to believe that a creature such as the platypus could be possible. The scientific discovery and examination of these creatures was unlikely, but far from impossible, as their existence demonstrated. The discoveries of such an unexpected creature led others to believe that bunyips could also exist. They tried to assert that the discovery of other unlikely creatures made the discovery of the bunyips more likely.

Though it is still possible that this is or at least was the case, the existence of bunyips remains incredibly unlikely. In fact, it is so unlikely that one could state that bunyips do not exist with a reasonable level of certainty. Many people have spent large amounts of money searching for mythical creatures. However, more monsters exist in our minds than could ever exist in the world.

For many years, information security and risk management has been an art rather than a science (Bernstein, 1996), with detrimental consequences. This has engendered a reliance on experts whose methodologies and results can vary widely and which have led to the growth of fear, uncertainty and doubt within the community. In place of searching for bunyips, it is preferable to implement systems that cover most failures and prepare to act when unusual events emerge.

The standard systems reliability engineering processes[xxviii] are applicable to information systems risk. These formulae and methods have been widely used in systems engineering, medicine and numerous other scientific fields for many years. The introduction of these methods into common use within risk and systems audit will allow the creation of more scientific processes that are repeatable and do not rely on the same individual for the delivery of the same results. Some failures will occur. A 99% confidence interval, though considered a good measure, still involves a level of uncertainty. But it is unwise to discard the normal occurrences in reaction to a black swan that may turn out to be something else again and not the problem it was seen to be. By assuming all black swans lead to catastrophic and unpredictable failure, we are again destroying the exception.

4.2.1 An investigation into the causes of system compromise

To test the causes of system compromises, I configured 640 Windows XP Professional systems that were on virtual hosts. The placement of each of the hosts was on an IP address external to a network firewall. Three separate tests formed the foundation of the experiment. For this, I set the baseline security of the system as a CIS (Centre for Internet Security) score. The CIS Windows XP Professional Benchmark v.2.0.1 (Shawgo, Whitney, & Faber) formed the security test metric.

These are:

1. A base install of a Windows XP SP 2 system,
2. An increasing CIS score was configured on the hosts,
3. A snort IDS was used to separate worms and other automated malware from interactive attacks.

Network traffic monitors were used to determine if a system had been compromised. The hosts had no third-party applications and initially had auto-updating disabled. A host, when compromised, was reset and reconfigured for the next set of survival tests. The reset utilised the VMware "snapshot" feature to take the system to a known good state.[xxix]

The SANS Institute (2005) defines a zero-day attack as when a "flaw in software code is discovered and code exploiting the flaw appears before a fix or patch is available". For this thesis, I define a zero-day attack as one that uses computer vulnerabilities that do not currently have a solution. This includes patching from the vendor, third party patches and workarounds. In this way, a vulnerability with a CVE number and third party protection (such as IPS filters or anti-malware updates that stop the attack) is not defined as a zero-day attack for this thesis.

This aligns with the definition of a "zero-day exploit" as occurring "when the exploit for the vulnerability is created before or on the same day as the vulnerability is learned about by the vendor" (Bradley).

4.2.1.1 Modelling the impact of a single control

The first test process separated the hosts into two classes: those with the Windows XP Firewall enabled, and those with the firewall disabled. No third-party products (including anti-malware software) were used on either class of system. Even with the release of Windows Vista and Windows 7, the same use and deployment of this control applies to both Windows 7 and Windows Vista. In addition, many organisations still use Windows XP.

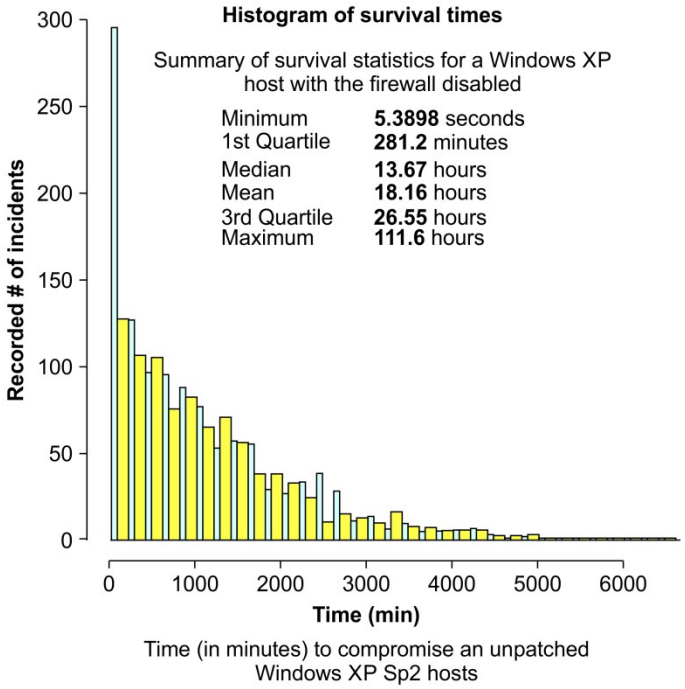

Figure 7. Survival time with the Windows firewall disabled.

The histogram in Figure 7 displays the distribution of survival times for the un-firewalled Windows XP hosts. The Conflicker worm that managed to compromise the un-firewalled hosts in quick succession skewed this result with the quickest time being 5.4 seconds from the network cable being connected to a scan that occurred (in May 2009). This was an exception and hence an outlier. The mean time to compromise of the hosts was just over 18 hours, with only 25% of the sample compromised in less than 3 hours.

When the results of the firewalled and un-firewalled hosts are compared, one can confidently assert that the Windows host firewall is a control that has a statistically significant effect when used. With the firewall enabled, the mean survival time of the Windows XP SP2 systems increased to 336 days. No system with this control enabled was compromised in less than 108 days. With the maximum survival time for an unpatched and un-firewalled Windows XP system predominantly measured at less than five days and the minimum compromise time at

108 days with the enabling of the firewall and no additional patching, it is hard not to conclude that the Windows Firewall makes a statistically significant difference to the security of the system (see Table 4).

Table 4 Survival times.

	Statistical Analysis of Survival times	
	Windows Firewall Enabled	Windows Firewall Disabled
Mean	18.157 Hours	8,064.640 Hours
	t = -170.75 df = 2272 p-value = 2.2 Exp -16	

This exercise used Snort IDS, which provided the details of the attacks allowing an analysis of worm (automated) compromises against manual (attacker, script kiddies, etc.). The IDS sat between the Internet connected router and the virtualised Windows XP hosts. Any outgoing traffic was investigated.

In the results of the 640 hosts that were used for this experiment, no system was compromised with a zero-day attack. Many new and novel attacks against known vulnerabilities did occur, but not a single compromise resulted from an unreported vulnerability. Further, no attack without a patch was used to compromise any of the systems. This means that if the systems had been patched, none of the attacks would have succeeded.

In a simple test of a single control, the enabling of this control had a marked effect on the survivability of the system. This demonstrates that leaving the host running with the firewall enabled provided a good level of protection (without a user on the system). This does not reflect a true Windows XP system as any third-party applications and user action have been introduced to confound the results. All connections to these hosts

are from external sources (such as a server model) to the host and no users are browsing malware-infected sites. In general, a Windows XP system will have a user and will act as a client. This introduces aspects of browsing and retrieving external files (e.g. email). These aspects of the host's security will change the survival rates of a system, but it is evident that there is a significant advantage from even a simple control (see Figure 8).

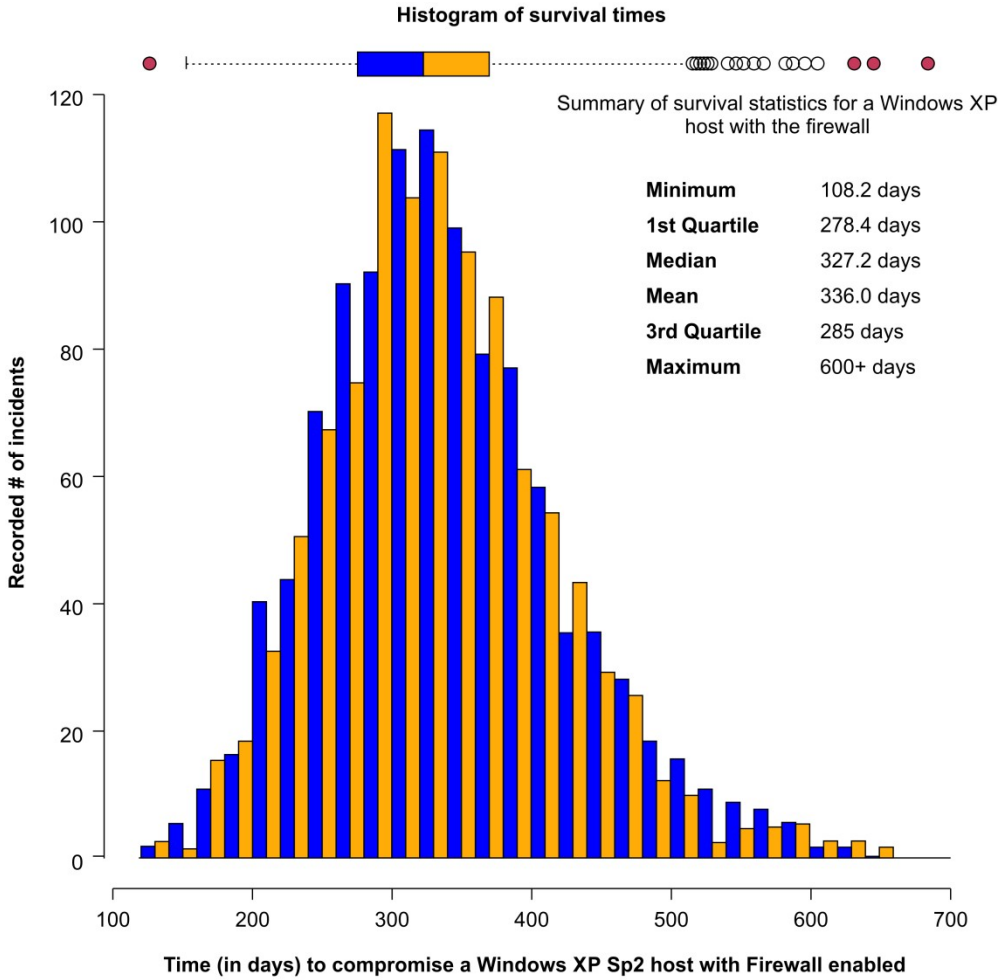

Figure 8. Survival time for Windows XP classified by interactive attacks and automated malware (worms).

In a sample of 136 home systems from corporate computers that have been tested and a sample of 231 systems inside various corporate networks, few systems were running a firewall. Of the hosts tested, 31.28% (or 23 systems) had the Windows XP Firewall or a commercial equivalent installed and running. Of the internal systems tested in this study, 6.1% had an internally (inside the corporate firewall) enabled firewall (14 hosts). The ability to enable IPSec and Group Policy within a corporate environment is a control that is generally overlooked or bypassed. Enabling (or rather not disabling) the Windows firewall produces a pronounced benefit to the survivability of systems.

This first experiment shows marked benefits from a simple control without the worry of any black swan effect.

4.2.1.2 Modelling system survival by attack class

The results of a series of hazard modelling experiments on Windows XP as previously presented were limited to a base Windows XP install. This was designed to test the effect of enabling or disabling the firewall. I next altered the experiment to investigate the attacks in classes; comparing the systems that have been compromised by an automated process (such as worms) against those which have at least some level of interaction.

The introduction with Windows XP SP2 of a firewall enabled by default is demonstrated to have had a significant impact on the overall security of networked systems. Each of the systems was reset and configured with a varying level of controls. The CIS metrics where calculated using the automated tool. Systems were distributed evenly between the metrics in 5% intervals (that is, 32 systems were allocated to each 5% bracket). The systems were made either more or less secure by

enabling and disabling controls until a complete spread of scores was created.

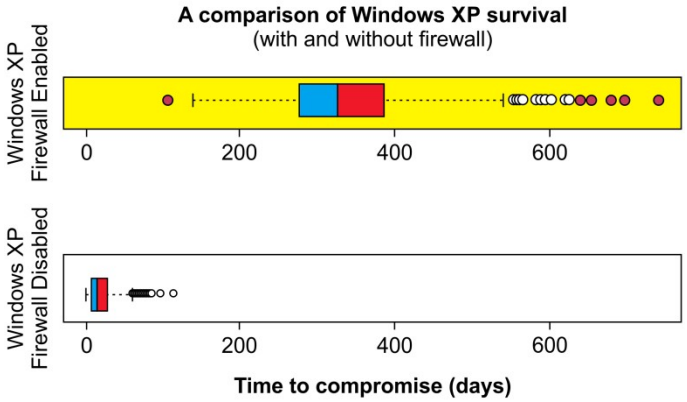

Figure 9. Comparing the use of the Firewall to an unprotected XP system.

Figure 10 displays a significant difference in the patterns of compromise due to automated and interactive attacks. It is evident from the plots that worms act faster against vulnerable systems and that interactive users (attackers) are more capable at compromising secure systems. This is more easily seen on an overlay plot (Figure 11). This displays a plot of the survival time against automated processes (green) overlaid with that of manual processes (red). The Loess fit for each is also incorporated into the plot.

Other results reveal that the more secure a system is (in this case, patched of known vulnerabilities), the more likely that a compromise is manually initiated. Likewise, the less secure (or patched and vulnerable) systems are exposed to more automated attacks (e.g. worms).

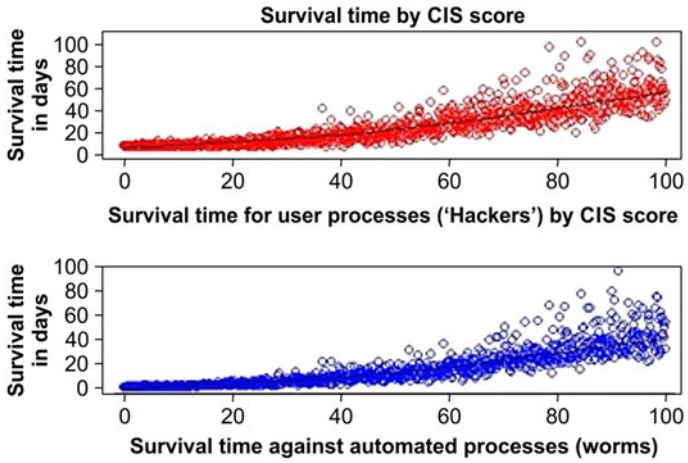

Figure 10. Survival time for Windows XP classified by interactive attacks and automated malware (worms).

4.2.1.3 System Modelling by CIS Metric

A selection of 48 Windows XP SP2 computers was used for a test that incorporated both 16 physical hosts and 32 virtual machines. This was conducted to examine the differences (if any) that may result with a virtualised host in place of a physical host. The tests were run over a 600-plus day period starting from November 2007. When a physical host was compromised, it was taken offline for 10 days. In this period, the host was rebuilt in a slightly different configuration. The 32 virtual hosts were built with differing levels of patching. These hosts have been reverted to a VM snapshot following a compromise. At this point, they would be re-patched and reassessed.

The same Snort IDS system used in the previous experiment was deployed to measure the attacks against the physical hosts. The 32 virtual hosts were configured on a single high-end Red Hat server running Snort. No filtering was conducted, but all attacks were logged. The survival time for the host is set as the time from when the host was placed as live on the network until a local compromise occurred. The 16 physical hosts where connected to a Cisco switch sitting behind a Redhat Linux host running Snort and acting as a forwarding router.

Each host in both the physical and virtual configuration was configured on a /29 network. This was assigned an internal IP in the 10.x.y.z address ranges with the Redhat host being assigned the lower IP address and the host being tested to the upper IP address. Static NAT was used to pass a real IP address to the host in the range of 203.X.Y.194 to 203.X.Y.242 with a netmask of 255.255.255.192. The full address was not reported to minimise any impact on ongoing experiments.

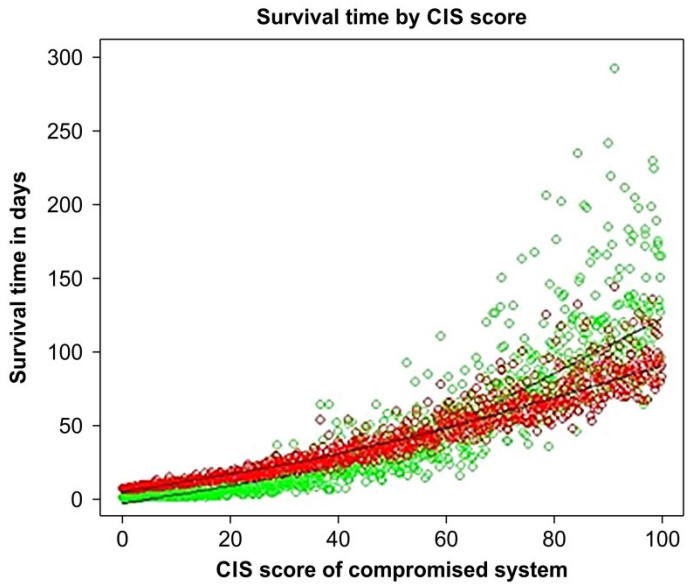

Figure 11. Automated vs interactive attacks and survival times.

The iptables configuration on the two Redhat systems was configured to allow any IPv4 traffic from the Internet and to block any IPv6 traffic. The Redhat host did not have a publicly routable IP address. Internet hosts could connect to any system on any port. The only restriction was designed to block traffic to and from the Windows XP hosts to any other host on the same network. This allowed the host to be compromised from the Internet but a compromised host could not see another host on the same network. The Windows XP firewall was disabled for all CIS scores less than 90 and for some hosts with scores greater than 90 (although it is

difficult to create a host with a score greater than 90 and the firewall disabled).

This was done to create a level of independence, forcing attackers to compromise systems from the same way without being able to “hop” across systems (as occurs in real compromises). The goal of this experiment was to record initial compromises and not the subsequent process (that being the goal of a separate and ongoing experiment). The times and measures have all been recorded and analysed. As before, no web browsing or other internal activity was conducted from the systems under test.

The scatterplot (Figure 12) is the plot of the measures score using the CIS scoring system against the time that it took to compromise the host. It is shown that there was a significant benefit in achieving a score of 80+. Any score of less than 40 was compromised relatively quickly. A score of 46 was compromised within 24 hours. All scores of 60+ remained uncompromised for at least a week. One host with a score of 59 on the CIS scale remained uncompromised for 98 days.

Similar results have been recorded for the hosts in the VM group (blue) and the physical group (red) in the scatter plot (Figure 12). A Loess best fit has been applied to this scatter plot, marking the expected survival time by CIS scoring. As the score increases, the variance also increases, but this can be modelled using a function of increasing survival times. No statistically significant differences in survival times have been noted because of the host being virtualised or physical (Figure 12).

From these results, one can assert that automated systems are more likely to compromise poorly configured systems than well-configured ones. This result is no more than common knowledge; however, it is also

apparent that an interactive attacker is more likely to succeed in compromising a well-configured system when compared to an automated process. It is also shown that even the best-configured system fails in time.

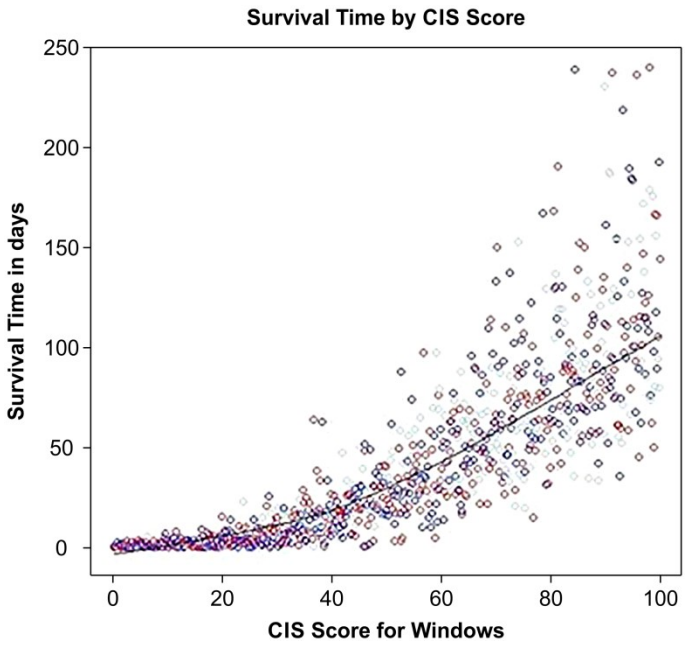

Figure 12. Survival for physical vs. virtual hosts.

Again, no system failed from an unknown attack. Of note was that several systems where compromised using new but known attacks. In the majority of attacks against a system with a CIS score of greater than 60 and with the Windows firewall enabled, the system was compromised between patch cycles. This involved the attack occurring against a new vulnerability before the scheduled patch release was due. Additionally, in all instances, these attacks involved systems that were not interactively managed. Workaround existed for all the incidents that led to compromise of the more secure systems. Further, more sophisticated anti-malware, firewall or other system security software would have stopped these attacks. Therefore, these attacks are not classified as zero-days. The

vendor did not have a public patch, but a workaround or third party control existed in all instances.

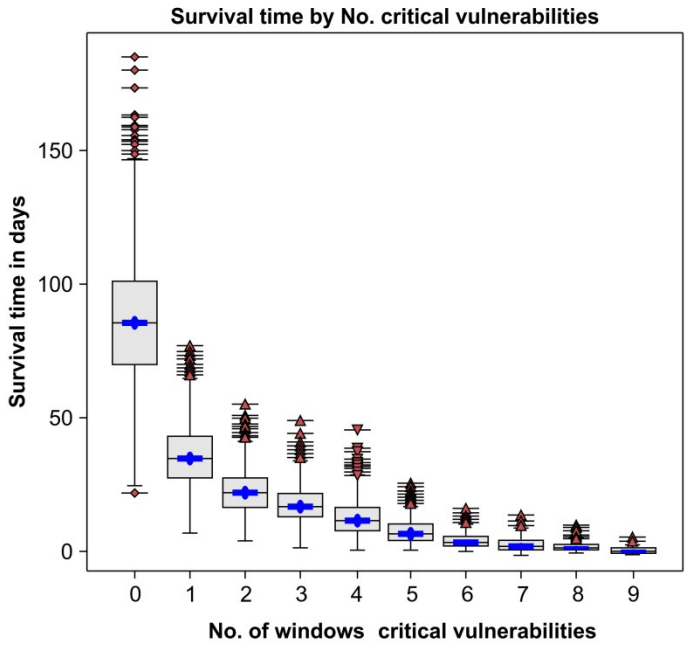

Figure 13. Mapping survival by critical vulnerabilities.

The issue comes down to economic allocation of scarce resources. Numerous solutions could have stopped all the attacks against the secured hosts. Some of these solutions would have cost less than implementing the controls that gave the Windows system a greater CIS score.

4.2.1.4 Mapping survival time against vulnerabilities

The next part of the experiment involved the configuration of 16 Windows XP SP2 hosts with a set and measured number of critical vulnerabilities. These hosts were left unpatched (ranging from 1 to 10 unpatched vulnerabilities per host) for a selected set of vulnerabilities. The experiment involved applying all other patches for newer vulnerabilities as they became available. The particular vulnerability was randomly selected on each host. Each vulnerability was selected from the SANS Top 20 vulnerability list (SANS, 2007).

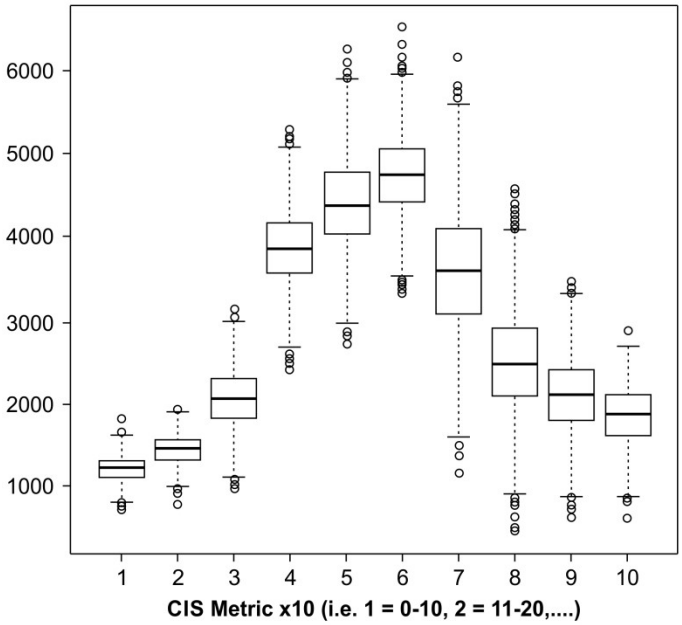

Figure 14. Attacker time by CIS metric.

All the hosts used virtualisation with “snapshots” enabled. A host that was compromised was reassigned with a new IP address and was reactivated 14 days later. Reactivation involved restoring the host to the snapshot and patching it. The host was left with the same number of critical vulnerabilities, but a different set of vulnerabilities was selected randomly from the SANS top 10 list.

The results of the experiment provided a useful model for predicting system survival. A system with a greater number of vulnerabilities is compromised quicker (Fig 14), a negative exponential relationship. Additional vulnerabilities exposed on a host significantly increase the likelihood of compromise. Hence, it can be asserted that the greater the number of vulnerabilities that a system has, the faster it is compromised. No system with six or more unpatched network accessible vulnerabilities remained uncompromised for more than 15 days. A compromise occurred in as little as four days on systems with two vulnerabilities. A system with no critical vulnerabilities can be expected to survive for several

months even without administrative interaction (Fig 15). Again, none of the attacks against these systems could be termed black swans. Each was known and predictable. In each case, known a workaround existed.

4.2.1.5 Attack Time by CIS Score

The time between the initial instigation of an attack until an attacker either moved on or compromised a host was analysed for related systems. Due to the lack of means to correlate between systems that an attacker may use, this value is lower in many cases than would be recorded if all the IP addresses used by a single attacker could be utilised. As such, this result is only indicative and does not take attacks from single attackers who use multiple addresses into account.

Figure 14 shows an inflection point on the amount of time spent attacking a system. More secure systems (a high CIS metric) would appear to discourage attackers where unsecure systems (a low CIS metric) are quickly compromised. Some attackers are determined and will continue to attack a host for extended periods in time. Even when an attacker receives little or no positive feedback, many persist in testing a system over time.

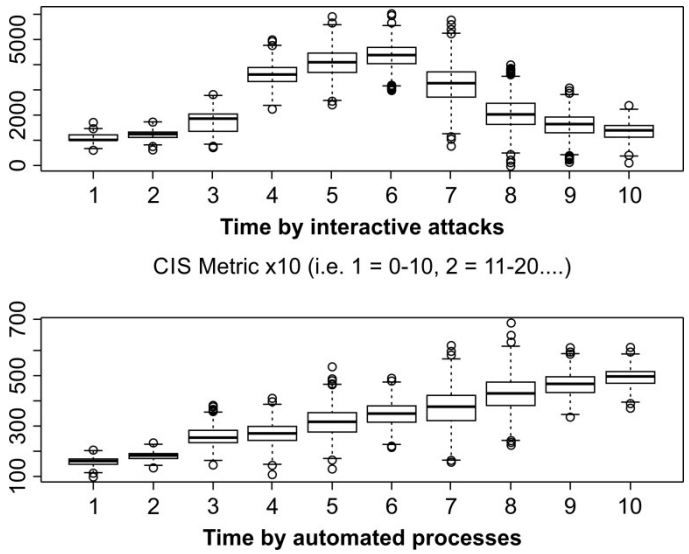

Figure 15. Attacker time by CIS metric and attack class.

This holds even more strongly when the attack class is separated by automated and interactive attacks (Figure 15). Automated attacks have a low amount of time on the system due to compromise. When a system is not compromised, it displays little intelligence into the attack patterns deployed against a host. An interactive attack displays a marked drop in the time per host as the security metric increases. As the host exhibits fewer vulnerabilities, the attacker spends less time exploring the host. It must be noted that the attackers are random in this experiment. The distribution of attackers is unlikely to contain dedicated attackers who are targeting a particular site (such as many cybercriminals and "hactivists" would do (Gordon & Ford, 2002). The hosts appear as normal corporate and home systems. No information was provided to the attacker that would enable them to associate the hosts with any particular organisation.

4.2.1.6 SANS Top 20 critical security controls[xxx]

As was seen above with the example of a simple control in the enabling of a Firewall by default in Windows XP, there are many easy ways to

implement controls that produce a significant positive benefit for little cost, making them an economically optimal approach.

The Sans top 20 critical security controls process provides such a methodology that can be used in helping to implement a set of security controls and to assess a systems security state. As can be seen from the other examples in this chapter, a focus on known proven attacks and controls to mitigate those attack vectors is economically effective whilst at the same time proving an environment with significantly lowered risk of compromise. Even if the methodology is imperfect and covers far less than a more comprehensive methodology (such as those from CIS), it gives the user a concrete method to significantly reduce risk with minimal costs.

4.2.2 Discussion

User interactions affect survival times in the real world. This will of course change the models produced by this experiment. The modelling of complete interactive systems was outside the scope of the experiment, but user actions can be modelled (Pillai & Kumar, 2007) with more advanced techniques (such as clustering algorithms). The results of the experiments presented demonstrate that the focus on zero-day or black swan events is misplaced. These can cause damage, but they are no more likely to damage a system than an attack using a well-known vulnerability that has not been patched. As Anderson (2001) notes, this is hard. The economic costs (Arora et al., 2006) of maintaining a system with all the required patches for all applications are frequently greater than the cost of purchasing and installing the software.

The problems with focusing on zero-day attacks are two-fold. First, the number of attacks that occur from true (Bednarski & Branson, 2004) zero-day events are fairly low. These incidents also cause the same

damage as an incident that has resulted from a known vulnerability; a system compromise is a system compromise, regardless of origin.

Next, and more importantly, the number of negatives is too large; there are simply too many black swans. For every attack that can succeed, there is a near infinite number of possible attack vectors, most of which never occur and will never occur. One such example is the oft-recurring argument as to the possibility of an attack against a fax server running on an analogue line.[xxxi] Much effort that could have been better applied to securing known issues has been applied to such bunyips. Even where zero day events occur as a platypus (that is, a completely unexpected event that few would have believed possible), the impact is rarely greater (Arora et al., 2006) than a compromise from an issue that was exposed but known.

As Lewis Carroll (1871) noted when satirising Victorian inventions, we change little and often give little thought to the economic allocation of funds to mitigate risk:

> "I was wondering what the mouse-trap was for." said Alice. "It isn't very likely there would be any mice on the horse's back."
>
> "Not very likely, perhaps," said the Knight; "but, if they do come, I don't choose to have them running all about."

A focus on the unknown at the expense of the basics is foolhardy at best. An organisation can expend effort addressing all possible and even unknown issues like Carroll's knight, but this will divert expenditure from those events with the greatest impact. By focusing on the unknown, organisations fail to address the issues that have the greatest impact (Varian, 2004b). The result of such an action is waste and loss (Friedman,

1953). Addressing the known issues also mitigates many of the unknown ones without trying.

4.3. A Comparative Study of Attacks Against Corporate IIS and Apache Web Servers

Broersma (2005) has suggested that Microsoft server software is more likely to be attacked than Linux due to perceived insecurities within these systems. Previous research has focused on investigating the trends[xxxii] against the underlying operating system. The purpose of this research was to investigate a single factor, namely, the Web server software as a vector for attack.

This project was not designed to test the relative strengths or security levels of either operating system, but rather to determine the relative attractiveness of each of these systems (the system being the combination of the web server and the underlying O/S) to an attacker.

In this experiment, the systems were configured to appear as a financial services organisation's client website. There were two systems, one running on Apache and the other on IIS. All other services had been firewalled and only access to the web server using HTTP (TCP port 80) and HTTPS (TCP port 443 over SSL) was allowed. The actual pages and data presented by the web servers were identical but for the server headers. Unlike previous research, the focus of the experiment was to record the volume of attacks from individual sources. By this process, I could answer the following questions:

1. Which web server software product (IIS or Apache) will have the most "vulnerability scans"[xxxiii]?

2. Which product will an attacker spend the most time with in an attempt to “break” or “crack” the software security settings?
3. What is the subsequent effect of hiding the host headers[xxxiv] on the servers?

4.3.1 Methodology used in the study

The research was based on a controlled trial with naturally randomised subjects. The honeypot design without notification allowed for the randomised discovery of the systems used in the tests. This naturally excluded targeted attacks from this study as the systems only simulated real banking sites, and had not been available for a enough time to be replicated in search engines.

A *robots.txt* file excluding all search engines and the Internet Archive was implemented to act as if the system was not allowing caching. This was added to stop Google scans and similar intelligence gathering methods as these could bias the experiment. This was designed in part to cover the fact that the real system was not available before the start of the experiment (Peisert, & Bishop, 2007).

By restricting access to the servers through a firewall to only the Web service on TCP ports 80 and 443, it was possible to demonstrate system attractiveness on a single defined service. The results of this experiment support the research efforts of the Honeynet Project,[xxxv] Symantec,[xxxvi] and the Internet Storm Centre.[xxxvii] In correlation to the prior research on this topic, it was initially confirmed that a greater number of attacks was made against the Windows server.

Rather than focus on the survivability of a host, this experiment served to determine the attractiveness of the host to an attacker. Unlike many previous experiments on this topic, this study was designed to test the

effect of obscuring the servers by hiding the host headers and information thus available to an attacker.

The valuable effect shown in this study was not that IIS on Windows was more attractive than Apache on Linux, but rather that Linux was less attractive to attackers. The reasons for this have not been determined conclusively, but LAMP-Based systems were less attractive than IIS, MSSQL and .Net based implementations. For this thesis, we have not tested the effects of Apache on Windows, which would make a valuable follow-up test.

The time and effort an attacker spends on a particular system cannot be used to determine the difficulty in attacking that system. Attractiveness is not the same as survivability and a follow-up test was conducted using a perceived vulnerability in both Windows and LAMP. The follow-up test involved configuring the system to appear as if it had a vulnerability as defined by a Nessus scan. A collection of IIS 7 and Apache vulnerabilities was simulated. These vulnerabilities were selected randomly from the CVE list and were simulated using HPING to modify the packets returned by both servers such that they mirrored the responses of a vulnerable system. The vulnerable versions of the software were not used as it would not be possible to control for individual responses (most vulnerable software versions can be attacked in several ways). This was used to differentiate attacks from worms and users. The traffic was filtered using SNORT to both record the data as well as to determine and monitor attacks.

4.3.1.1 Description of experimental study

Subjects (i.e. the attackers) discovered the systems based on their own activities. As there is no way to attract subjects to the systems, it was expected that a random sample of the population of “hackers” on the

Internet would find and explore the system at any given time. No details of the source of the attacks were analysed.

The analysis took the nature of the probes into account as well as the relative amount of time spent on each system.

4.3.1.2 Experimental procedure

The data collected from this experiment is based on two "Honeynets" deployed to appear as the primary and DR site for a fictitious financial organisation. Each Honeynet consisted of two servers configured to run VMWare. Each of the VMWare servers was configured to run a Web server using bridged network access.

The first virtual server was configured using Red Hat Enterprise Linux 6.0 and Apache version 2.2. The second virtual server consisted of a Windows 2008 server system running IIS 7.0. Each of the pages was configured to appear as a client portal in our fictitious financial services organisation (Figure 16).

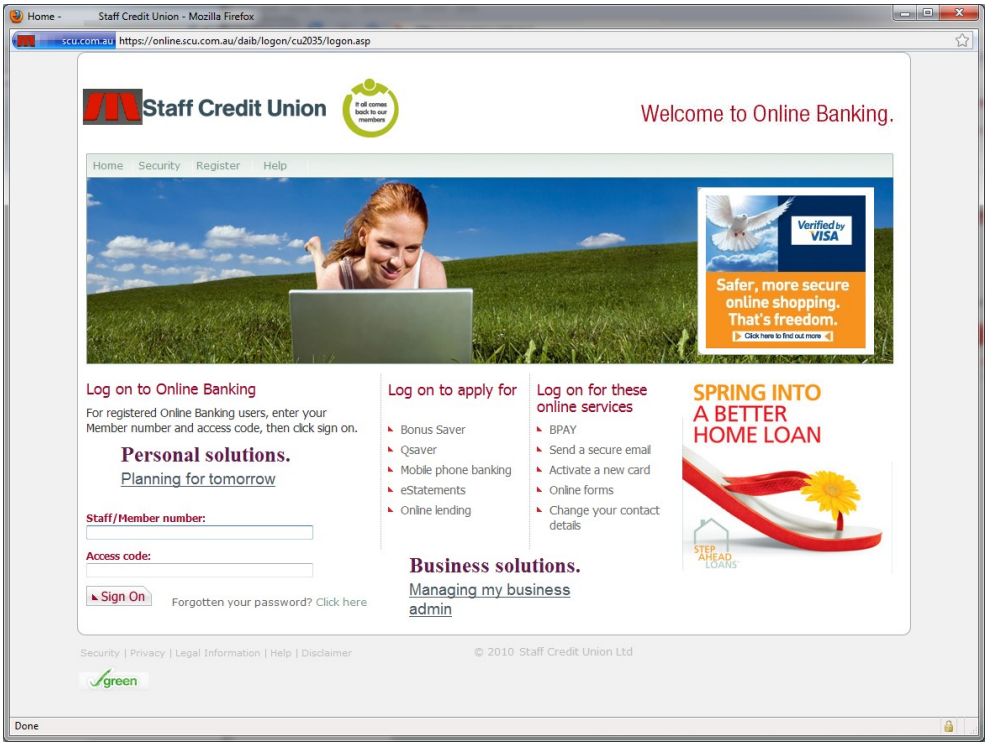

Figure 16. The "Staff Credit Union" banking login page.

This organisation was created through an amalgamation of existing credit union and banking sites and was designed to look and feel as authentic as possible. The domain was not formally registered, but a DNS zone was created on the local server. This zone and domain were never published nor have they been advertised.

To simulate a financial services organisation, these systems were installed behind a firewall, which only allowed access to TCP port 80 and a simulated access to TCP port 443 when authenticated. The Honeynet was linked to "real"[xxxviii] servers from a fictitious organisation. Both systems were configured to require authenticated access before allowing any access to a backend application.

Using the Snort IDS software, the number of attacks and thus the effort expended by an attacker on each server was measured and

recorded. The open source IDS product, SNORT, was used to collect the data. SNORT was installed on the underlying operating system that hosted the virtual machines as well as running the IPTables firewalls. A separate monitoring server was run on a Spanned switch port. The underlying system was Redhat Enterprise Linux 6.0. The Redhat system was configured with an IP address that could not be routed to the Internet and which was monitored for access (no access or compromise was noted in this system). In this manner, it was not possible to detect the IDS deployment from the web server. All systems were patched fully before being deployed.

There were two phases to the first stage of the experiment with the first phase involving leaving the web host headers unaltered. The second phase involved hiding the web host headers. By this, the server was stopped from advertising its software version. In the first phase, the systems responded as Apache version 2.2 and IIS 7.0. In the second phase, both systems were configured to respond with "Secure Web Server version 9.3". No other changes were made and a determined attacker using P0F or NMap could have differentiated the IP packet information to determine a difference between a Linux or Windows system (as a guess).

The results were collected daily for the duration of the test.[xxxix] Informational data (that which only provided details on the system and did not lead to a vulnerability) were excluded from the results. All attacks detected by SNORT were collected together and no effort has been made to correlate the levels of attack against each server. The monitoring server was used to collect all network traffic to and from the servers. The switch was spanned on the Monitoring Server port and TcpDump was set to capture all traffic. This was set as both a backup for the SNORT systems

as well as a means of collecting any attacks that had been made against systems that could have bypassed SNORT.

Future research and analysis to correlate the levels of attacks and statistically analyse the intensity of attacks against each server as a function of time is planned.

4.3.1.3 Steps to physically control variation

To minimise variation, the Honeynet servers were configured as follows (Fig 17);

- Both systems were installed on matching hardware and domain addresses,
- Both systems were booted from a Live DVD base created using RHEL in the manner of a Knoppix distribution,
- Both systems resided on the same switched network and were active at the same times,
- The IDS system was not available or visible to the external network,
- Results were randomised as the systems were not “advertised,” and it is expected that they were accessed by general network scans and probes,
- The IP addresses of the systems were sequentially allocated such that a probe could detect both at the same time.

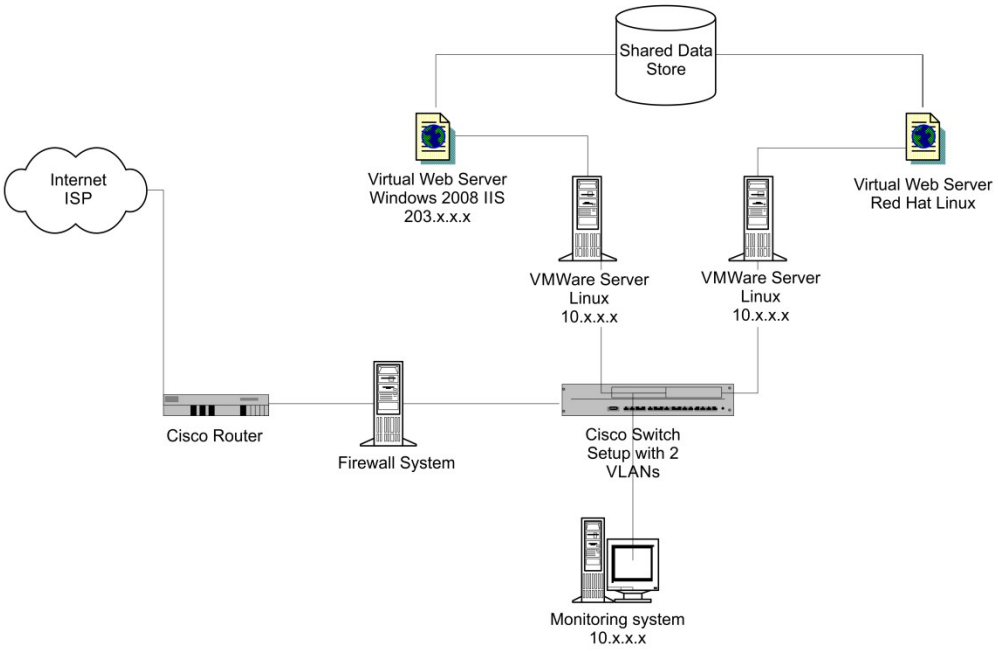

Figure 17. Network diagram.

4.3.1.4 Steps to statistically control variation

When either system was attacked by a DoS or DDoS attack, both systems were made unavailable using IPTables. SNORT was configured to trip a default block rule that took effect on both systems for any IP address that was noted as attempting a DoS or DDoS by Snort. This was done as the test was not designed to determine DoS/DDoS situations and it was decided that it was better to not continue to record data on one system while the other was not being tested.

4.3.2 Results and Discussion

The data was subsequently analysed based on the average number of individual source and attacking hosts of each system and the number of individual attacks registered per host. If an attacker had been attacking from multiple systems (such as from the use of a botnet), this was not determined and each attacking host was included individually.

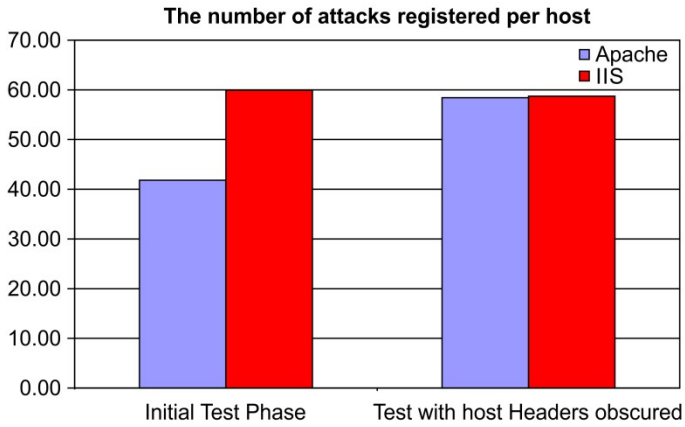

Figure 18. Phase 1 results of attacks against IIS and Apache (attacks / day)

The test was stopped following the collection of data for Phase 1 of the test and the systems were no longer routed for a period of six weeks before a new IP address in the same C-Class range was allocated and the system was redeployed using a simulated and randomly selected vulnerability from the list of vulnerabilities. Vulnerabilities were selected randomly from the CVE list.[xl]

The Honeypots were deployed using the methodology detailed by Bednarski & Branson (2004). The systems were left running with the traffic being recorded to collect data for several weeks with the Web servers host headers remaining visible (Phase 1). Next, the servers were reconfigured to each display an alternate host header; "Secure Web Server version 2.3" (Phase 2). This was designed to obscure the system details from a potential attacker.[xli]

The results of these tests are displayed in Figure 18 and reveal no statistical differences between attacks against the hosts when the host headers are obscured. In Table 5, the mean amount of time an attacker spent in attacking the web service is displayed for CVE vulnerabilities. The high and medium level ratings are determined using the CVSS[xlii] rating.

Table 5 Mean Time Spent Attacking with Vulnerabilities (Seconds)

IIS		**Apache**	
Medium	High	Medium	High
1090.1	2097.0	1242.5	1272.9

In Phase 2, the average time per attacking IP address was recorded for both the simulated medium and high level attacks, again demonstrating a clear distinction between IIS and Apache with attackers spending more time on IIS high level attacks (2097.0 seconds) than Apache high level attacks (1272.9 seconds). Phases are separated depending on if the server headers have been changed or not.[xliii] These results demonstrate that an attacker will extend significant effort against an IIS based system with a perceived high level vulnerability (Fig 19).

It would also be possible to posit an alternative explanation that the mean time does not necessarily show a preference, but also "difficulty" in exploiting the target. No evidence has been found that attacking IIS is significantly more difficult than attacking Apache (or for that matter that Linux is less difficult than Windows to attack) and it is more likely that the difficulty of an attack is related to the individual vulnerability and not the underlying system.

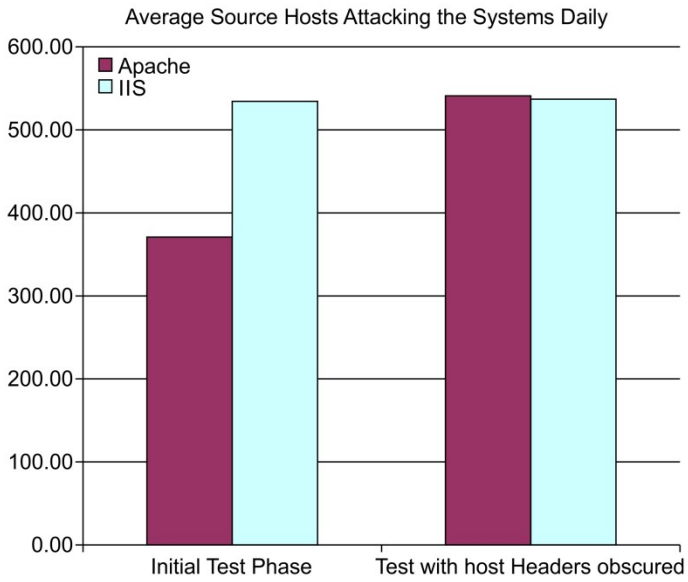

Figure 19. Phase 1 results of attacks against IIS and Apache.

In addition, the economics of an attack would likely lead to seeking the easier target if this was the case. In such an event, it would be logical to posit that given a choice of a Linux or Windows based system, if Windows was indeed "more difficult" to attack, a clear preference for Linux would be expressed with a lower time expended on Windows systems expressing the same data.

The results of the experiment clearly demonstrate a similarity in the results obtained when the server type either is unknown or is determined to be a Microsoft Windows system. However, there was a markedly lower intensity and volume of attacks against the Apache Web server when its host headers were displayed.

To determine whether the Microsoft Windows IIS Web server or the Apache Linux Web server would attract more scans or attacks than its counterpart, a two-sample t-test was performed on the number of source hosts detected per day (Figure 19). When choosing a null hypothesis (H_o) that there is no difference between the Apache or IIS Web server, it was

found that the results of the initial phase of the experiment were significantly different ($t = 29.59$, $df = 54$, $p < 0.1507$) at the alpha = 20 level.[xliv] Therefore, the null was rejected in favour of the alternative hypothesis H_A: that there was indeed a difference between the servers.

Table 6 Mean Attacks by Day

Mean Daily Results	Apache		IIS	
	Number of attacks detected per host	Number of Source Hosts Detected Per Day	Number of attacks detected per host	Number of Source Hosts Detected Per Day
Initial Test Phase	41.82	370.71	59.93	534.36
Test with host Headers obscured	58.43	540.86	59.54	536.93

The results support the hypothesis that the Apache Web server on Linux is less likely to be attacked than IIS Web server on Microsoft Windows 2008.

An ANOVA analysis of the results of the subsequent tests demonstrates a significant difference ($F=5.4402$; $df = 3$, $p < 0.0019$) in the intensity of the attacks. It can clearly be seen in Figure 18 that an attacker stops an attack with less effort when it has been determined that they are attacking Apache on Linux.

A comparison of the number of attacks detected per host for both the obscured IIS server, and either server with the host headers altered, demonstrated no significant difference ($F=0.0007$; $df = 2$, $p=0.9993$).

Again, the null hypothesis (H_o) that there is no difference between the server groups tested is rejected. Means comparisons for all pairs using Tukey-Kramer HSD at alpha = 5 show that Apache Web servers are less likely to be attacked.

ANOVA analysis of attacks by source hosts when the header was not Apache on Linux demonstrated no significant difference (F=0.0007; df = 2; p=0.9993).

ANOVA again supports the assertion that there is no significant variation (F=0.0344; p=0.8538, RSquare = 0.000859) when comparing the results of the Phase 1 tests against IIS to the Phase 2 tests with the host headers obscured.[xlv]

Conversely an analysis by ANOVA of the Phase 1 tests against Apache to the Phase 2 tests with the host headers obscured significantly (F=5.7659; p=0.0211, df =3) supports the claim that attackers are less likely to attack Apache on Linux.

Further, when these results are coupled with the initial analysis (F=14.4513; p=0.0004, df = 3) of attacks against Apache vs. IIS from above, it is plainly evident that there is support for the assertion that an attacker does not care what the server is if it is not Linux.[xlvi]

These results would suggest that the threat against Internet deployed hosts is moving from automated scanning tools to more human-intensive processes. By specifically avoiding the Apache Linux system (when not obscured), there is evidence to support the contention that attackers are manually targeting systems and actively stopping attacks they deem to be "too difficult".

4.3.3 Limitations of this Study

No effort was made to analyse the levels of attacks against any server. It may be that more high-level attacks are made against a Linux server for example; this assertion has not been tested. In this study, all levels of

attack were treated equally, whether they were designated as a low, medium or high-level attack.

As noted above, an alternative explanation would be that the mean time does not necessarily show a preference, but does show "difficulty" in exploiting the target. It would be necessary to create a separate experiment to collect evidence as to whether attacking IIS is significantly more difficult than attacking Apache (or for that matter that Linux is less difficult than Windows to attack) and it is more likely that the difficulty of an attack is related to the individual vulnerability and not the underlying system.

4.3.4 Future extensions to this experiment

Some potential areas of further research have emerged from this study. An attacker will avoid Linux servers that are not obscured, though this study can provide no reasons for this behaviour. A valuable test would be to use Apache installed on a Windows server. That could be used to show if attackers avoid Apache or Linux.

It is suggested that researchers consider this study and its conclusions as an initial exploration into the methodology of an attacker. Research into the motivations driving this behaviour in an attacker needs to be conducted. Further research is essential to develop appropriate strategies and measures to secure systems sufficiently. It is essential to understand the psychology of the attacker if effective controls are to be developed and deployed. A study where the host headers on a Microsoft Windows IIS host are altered to simulate Apache on Linux could also shed further light on the question.

This study has shown that attackers are not so much attracted to Windows, but rather shy away from Linux-based systems. This may be

attributable to the increased W32 market penetration. Another possible reason could stem from a perceived greater level of security with Linux hosts. The results of this study do not demonstrate that either Linux or Microsoft Windows is more secure. However, the results do support the claim that attackers *believe* that Linux is more secure (and therefore less vulnerable to attack). As the average attacker was willing to spend a greater amount of effort to compromise a high vulnerability IIS Windows system when defined by the amount of time spent attacking the system (Table 5), it is apparent that IIS has become a more attractive target for attackers than LAMP based systems running the same data.

Further research is needed on this topic to determine *why* Linux is less attractive than Windows to attackers. In addition, experiments into the effects of using other systems (such as the MAC OS) could be conducted.

4.4. Aligning Systems Engineering and Information Security Risk

This chapter discusses the major methods used in risk measurement and audit, and extends this into other processes that are used within systems engineering (Elliott et al., 2000). Risk assessment is fundamental to the security of any organisation (Grandell, 1991). It is essential in ensuring that controls and expenditure are fully commensurate with the risks to which the organisation is exposed. The chapter starts with defining risk and related terms before proceeding into the methods used.

The chapter defines processes as the methods that are utilised to achieve certain objectives. To implement and maintain a system, it is important to know precisely how these processes are implemented within an organisation. An objective, on the other hand, is a goal or something that people desire to have accomplished. It is important to ask just who

sets these objectives and how they are designed if risk management solutions are to be achieved effectively and economically.

Controls are the mechanisms through which an individual or group's goals are achieved. Controls are useless if they are not effective. As such, it is important to ensure that any control that is implemented is both effective and justifiable in economic terms. Controls are the countermeasures for vulnerabilities but they need to be economically viable to be effective. There are four types:

1. Deterrent controls reduce the likelihood of a deliberate attack,
2. Preventative controls protect vulnerabilities and make an attack unsuccessful or reduce its impact,
3. Corrective controls reduce the effect of an attack,
4. Detective controls discover potential (attempted) or successful attacks and trigger preventative or corrective controls.

4.4.1.1 Identifying risk.

A risk analysis consists of several stages, including threat analysis, vulnerability analysis, business impact analysis, and likelihood analysis (the probability of an event). A risk management plan (Wright, 2008) should consist of:

- Analysing individual risks based on the impact of the threats and vulnerabilities that have been identified from the risks,
- Rating the individual risks from highest to lowest importance,
- Creating a risk treatment plan that categorises each of the threats and vulnerabilities in order of its priority to the organisation, together with some possible controls.

4.4.1.2 Monte Carlo method

Several stochastic techniques have been developed to aid in the risk management process. These are based on complex mathematical models that use stochastically generated random values to compute likelihood and other ratios for our analysis model (Corcuera et al., 2004).

Monte Carlo methods[xlvii] can aid in other risk methodologies such as time-based analysis (Curtis et al., 2001). This technique further allows for the determination of the range of possible outcomes and delivers a normalised distribution of probabilities for likelihood. Combining stochastic techniques with Bayesian probability and complex time series analysis techniques such as heteroscedastic mapping is mathematically complex, but can aid in situations where accuracy is crucial (Dellacherie, 1982).

These quantitative methods help predict any realistic detection, response and exposure time in a manner that can be differentiated by the type or class of attack (Lui, 2011; Ni, 2010). However, these methods are generally more expensive than the other methods, requiring highly skilled specialists that an organisation may not be able to afford.

4.4.2 System Survival

When assessing network reliability, it is necessary to model the various access paths and survival times for not only each system, but for each path to the system. This requires the calculation of the following quantitative fields:

- $R(t)$ The Reliability function
- MTBF Mean Time Between Failures
- MTTF Mean Time to Repair/Fix

- λ The expected survival time

Other measures will be introduced later. Where possible, the standard systems reliability engineering terms have been used. In the case of a measure such as the MTTF, this represents the time both to discover and recover a compromised system. The true value estimate for the system comes as a measure of the applications on the system; this may be estimated for a less economically expensive (though less accurate) estimate. In this calculation, the compromise measure, MTBF, is best thought of as the mean time to the first failure.

This can be modelled with redundancy in the design. Here, each system is a parallel addition to the model. Where a system is required to pass another, a serial measure is added; for instance, if an attacker would be required to:

- bypass system A (the firewall) to
- compromise system B (an authentication server), which allows
- an attack against several DMZ servers (C, D and E), where
- systems C and D are connected to the database through
- a secondary firewall (system F) to
- the database server G (as displayed in Figure 20).

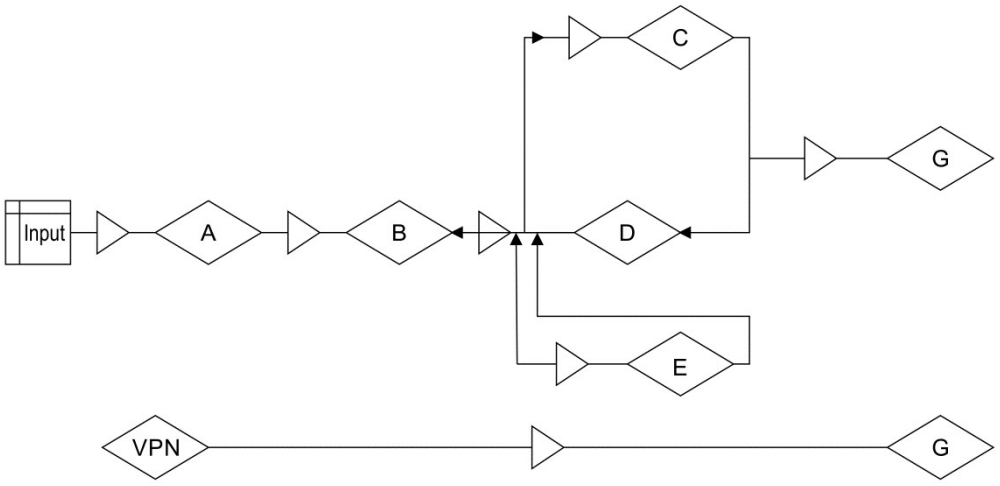

Figure 20. Attacking a series of systems.

The attacker can either attack the system directly through the VPN or by following the attack path indicated by the systems. If the firewall system A restricted the attacker to a few IP addresses, the attacker may perform one of several actions in attacking this system (to gain access as if the attacker was one of these IPs):

- Compromise the input host,
- Spoof an address between the input IP address (such as through a router compromise at an ISP or other system),
- Compromise the VPN.

Other options, such as spoofing an address without acting as a MITM, will leave the possibility of some attacks that cannot result in a compromise of system G. These could have an economic impact (such as a DDoS on the server) that would be calculated separately and could lead to a commercial loss.

Hence, the effective attack paths are:

- Input, A, B, C, F, G;
- Input, A, B, D, F, G;
- Input, A, B, E, C, F, G;
- Input, A, B, E, D, F, G;
- VPN, G.

In this instance, it is necessary to calculate conditional probabilities as these paths are not independent. Here the options to consider first include (using the term I to define an attack on the Input system and S to refer to a spoofed attack of the input system):

- The conditional probability of compromising system A given a successful spoof attack on the Input system, $P(I \cup A_1) = P(I).P(A_1 \mid I)$ (where A_1 refers to an attack on system A using path No. 1, or Input, A, B, C, F, G),
- The conditional probability of attacking system A ,
- The probability of attacking system G, $P(V \cup G_5) = P(V).P(G_5 \mid V)$.

Each of the attack paths can be treated as independent. Hence, the overall probability of an attack is the sum of the conditional probabilities from each attack path. Consequently, the attacker will most likely enter via the least costly path, but the probabilistic result allows for an attack from any path. The high and low probability attack measures are jointly incorporated in the process.

Presuming no other paths (such as internal attacks), it is feasible to model the alternate probability as not possible (or at least feasible). Here, $P_6 = 0 = \in$. Additionally, the probability of an attack over path 5 (the VPN) can be readily calculated without further input as:

$$P_5 = P(V \cap G_5) = P(V).P(G_5 \mid V)$$

Here:

$$P(V) = e^{-\lambda_V t} + \left(-\lambda_V t\right) e^{-\lambda_V t}$$ and

$$P(G_5) = e^{-\lambda_G t} + \left(-\lambda_G t\right) e^{-\lambda_G t}$$
$$P(G_5 \mid V) = \prod P(G_5)$$

Here there exists a single system behind the VPN. Where more than one system exists, it is necessary to calculate the joint probability as is detailed below. In the example, with only a single system:

$$P(G_5 \mid V) = \prod P(G_5) = P(G_5) \times 1$$
$$\therefore P(G_5 \mid V) \to P(G_5)$$ Equation (22)

Equation 23 holds as the probability of the attacker compromising system G when the VPN that has been compromised approaches 1. This is as the attacker has a single target with the VPN and the utility of attacking the VPN and no more is negated as no other systems exist and the VPN offers no other utility for the attacker alone.

The values, λ_V and λ_G are the expected survival time or the mean time to compromise for the VPN and database respectively as configured and *t* is the amount of time that has passed from install and represents the current survival time of the system (Jeanblanc & Valchev, 2005).

Here:

$$P_5 = P(V \cap G_5) = P(V).P(G_5 \mid V)$$
$$P(G_5 \mid V) =$$
$$= e^{-(\lambda_G + \lambda_V)t} - (\lambda_G + \lambda_V)te^{-(\lambda_G + \lambda_V)t}$$

Equation (23)

On the other hand, the probability of compromise to system I is based on the number of systems and as $n_I \to \infty, P(I) \to 1$. Basically, as more systems can connect to system A, the closer the probability of compromise tends towards a value of P=1. That is, as the number of systems susceptible to compromise increases, the probability of compromise approaches certainty. Where there is only a limited number of systems, the probability can be computed as a sum of the systems. Where there are many systems with equivalent (or at least similar), these can be calculated through the sum of the systems. If, in the above example, system E is replaced with a series of systems (each with the same configuration), it is possible to calculate the probability of a compromise of one of the "E" systems as follows:

$$P(E) = R(E) = 1 - \prod_{i=1}^{n} \left[1 - P\left(E_i\right)\right]$$

Equation (24)

Here, P(E) is a multiplicative and not additive function. As such, if system "E" is defined as a DNS server with a single BIND service and SSH for management of the host, an attacker has two means to compromising the system;

- Attack SSH, and
- Attack BIND.

The probability can be considered as independent in this case if there are no restrictions. In the example, DNS is an open service, that is, P(I)=1. The SSH service may or may not be open and could be restricted.

If this is the case, then 0<P(I)<1. In the simple case where no restrictions have been imposed on SSH, the probability can be calculated as a standard independent probability formula:

$$P(SSH) = e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}$$
$$P(DNS) = e^{-\lambda_{DNS}t} + \left(-\lambda_{DNS}t\right)e^{-\lambda_{DNS}t}$$
$$P(E) = 1 - P(SSH).P(DNS)$$
$$\therefore P(E) = 1 - \left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + \left(\lambda_{DNS}\lambda_{SSH}\right)t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} + e^{-\lambda_{SSH}t}\left(-\lambda_{DNS}t\right)e^{-\lambda_{DNS}t} + e^{-\lambda_{DNS}t}\left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}\right]$$
$$P(E) = 1 - \left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + \left(\lambda_{DNS}\lambda_{SSH}\right)t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - \left(\lambda_{DNS}t\right)e^{-(\lambda_{SSH}+\lambda_{DNS})t} - \left(\lambda_{SSh}t\right)e^{-(\lambda_{SSH}+\lambda_{DNS})t}\right]$$
$$P(E) = 1 - \left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + \left(\lambda_{DNS}\lambda_{SSH}\right)t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - \left(\lambda_{DNS} - \lambda_{SSh}\right)te^{-(\lambda_{SSH}+\lambda_{DNS})t}\right]$$

Equation (25)

The complication comes where one of the services has been restricted. This is a combination of the probability of compromising the restrictions on the service (that is spoofing or otherwise bypassing IP address controls) and the compromise of the service itself. This can be represented by:

$$P(SSH) = \left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}\right].P(I)$$
$$\therefore P(SSH) = \left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}\right].\left(1 - \prod_{i=1}^{n}\left[1 - P\left(I_i\right)\right]\right)$$
$$= \left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}\right] - \left(\prod_{i=1}^{n}\left[1 - P\left(I_i\right)\right]\right).\left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}\right]$$

In this case, there exists a probability $P(I) = 1 - \prod_{i=1}^{n}\left[1 - P\left(I_i\right)\right]$ where the allowed source systems (I) are limited to a total of "n" IP addresses (or keys). The probability $P\left(I_i\right)$ of any source system being compromised will vary, but may be estimated based on the type and location of each system. As more systems are added into the equation, the polynomial equation becomes more complex. If similar systems are also accessing this, these can be calculated and the equation simplified.

For example, if two (2) classes of systems exist (Linux and Windows Vista) that comprise the set of systems I_i for a total of 4 systems (2x Windows and 2x Linux) these can be defined using:

$$P(I_1) \simeq P(I_2) = e^{-\lambda_{Win}t} + \left(-\lambda_{Win}t\right)e^{-\lambda_{Win}t}$$

&

$$P(I_3) \simeq P(I_4) = e^{-\lambda_{Linux}t} + \left(-\lambda_{Linux}t\right)e^{-\lambda_{Linux}t}$$

In the case where $P(I_1) \simeq P(I_2) = 0.25$ and $P(I_3) \simeq P(I_4) = 0.2$ due to the system configurations and patch status, it is possible to calculate P(I):

$$\begin{aligned} P(I) &= \left(1 - \prod_{i=1}^{n}\left[1 - P\left(I_i\right)\right]\right) \\ &= 1 - \left[(1-0.25)^2.(1-0.2)^2\right] \\ &= 1 - \left[(0.75)^2.(0.8)^2\right] = 1 - \left[0.5625 \times 0.64\right] = 1 - 0.36 \\ &= 0.64 \end{aligned}$$

Equation (26)

In this case, the probability of a compromise due to SSH would become:

$$\begin{aligned} P(SSH) &= \left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}\right].P(I) \\ &= 0.64\left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}\right] \end{aligned}$$

Equation (27)

With the details from the example above it is possible to calculate the survival function for system E:

$$P(E) = 1 - 0.64\left[P(SSH).P(DNS)\right]$$

$$\therefore P(E) = 1 - 0.64\left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + \left(\lambda_{DNS}\lambda_{SSH}\right)t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - \left(\lambda_{DNS} - \lambda_{SSh}\right)te^{-(\lambda_{SSH}+\lambda_{DNS})t}\right]$$

Equation (28)

Thus, there exists a method to calculate the probability of each system as well as the conditional probability of that system.

The addition of a device (such as an IDS) changes or otherwise affects *t* and adds additional complexity to the calculations. An IDS, for instance, can limit the value of *t* through a probabilistic feedback process. The more effective the IDS is, the quicker an attack or another incident will be intercepted. In this instance, *t* becomes a probabilistic function based on how effective the IDS itself is. This becomes a combination of the following factors:

- The inherent accuracy of the IDS (which is a trade-off between TYPE I and TYPE II errors and it is a cost function in itself[xlviii]),
- The missed detection rate (even where an incident is noted, the analyst may miss the detection. As more false negatives are seen, the missed detection rate increases (Ikeda & Watanabe, 1962). Thus, increasing false negatives to capture all possible attacks ends in a limit where the IDS is no longer effective).

If the IDS does not detect the attack, the function mirrors that of the system without the IDS. Thus, the addition of an IDS is a limiting function. An increase in cost adds to the power of the IDS. That is, more analyst time and more detection capability lowers the false negative and false positive rate through an increase in cost. Each IDS system has an expected TYPE I and TYPE II error rate that will vary as the system is tuned to a precise environment. This yields an individualistic function for the organisation that can only be generally approximated for other organisations (even when the same IDS product is deployed).

For a given probability of survival, it is possible to calculate the expected survival time (t) of the system. This process becomes computationally infeasible in large systems with numerous inputs. For instance, on system E (as defined in Equation 23), it is feasible to rearrange the equation of the expected probability of system E being

compromised. If a calculation for the expected function of survival time for a set survival probability P is desired, rearrange the equations in Equation 23 as follows.

$$P(E) = 1 - 0.64\left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + (\lambda_{DNS}\lambda_{SSH})t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - (\lambda_{DNS} - \lambda_{SSh})te^{-(\lambda_{SSH}+\lambda_{DNS})t} \right]$$
$$\therefore if\ P = 0.99, \quad 0.99 = 1 - 0.64\left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + (\lambda_{DNS}\lambda_{SSH})t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - (\lambda_{DNS} - \lambda_{SSh})te^{-(\lambda_{SSH}+\lambda_{DNS})t} \right]$$
$$or \quad e^{-(\lambda_{SSH}+\lambda_{DNS})t} + (\lambda_{DNS}\lambda_{SSH})t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - (\lambda_{DNS} - \lambda_{SSh})te^{-(\lambda_{SSH}+\lambda_{DNS})t} = 0.0015625$$
$$\ln(e^{-(\lambda_{SSH}+\lambda_{DNS})t} + (\lambda_{DNS}\lambda_{SSH})t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - (\lambda_{DNS} - \lambda_{SSh})te^{-(\lambda_{SSH}+\lambda_{DNS})t}) = -6.4615$$

Equation (29)

This result is in the form of:

$$At + B\ln(t) = C$$

From Equation 28, it is clearly seen that as $t \to \infty$ $\left[t + \ln(t)\right] \xrightarrow{t\to\infty} t$. From these equations, if t is large, an approximation can be deployed to obtain a lower limit estimate of $At + B\ln(t) = C$ as $At \simeq C$. As such, an approximate for the lower limit of time for system E's survival is defined as:

$$t = \frac{6.4615 + \ln(\lambda_{DNS} + \lambda_{SSH})}{2(\lambda_{SSH} + \lambda_{DNS})}$$

Equation (30)

In Equation 24, it is demonstrated that the lower the value of t, the greater the error. Measuring t in seconds and substituting normal system values of λ allows for the use of Monte Carlo simulations to approximate the expected value of t.

For simplicity, let R represent reliability and Q the unreliability (hence, $1 - R = Q$).

For each application, there exists a possibility to use Bayes' theorem[xlix] to model the number of vulnerabilities and the associated risk. For open ports, the person evaluating the risk can use the expected reliability of the software together with the expected risk of each individual vulnerability to model the expected risk of the application. For instance, it is conceivable to model $P(SSH)$ using this method.

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Equation (31)

alternatively;

$$P(A \cap B) = P(B)P(A \mid B) = P(A)P(B \mid A)$$

Over time, as vulnerabilities are uncovered and fixed (if new vulnerabilities have not been introduced), fewer issues will remain. Hence, the confidence in the software product increases. This also means that mathematical observations can be used to produce better estimates of the number of software vulnerabilities as more are uncovered.

It is thus possible to observe the time that elapses (Guo et al., 2005) since the last discovery of a vulnerability. This value is dependent upon the number of vulnerabilities in the system and the number of users of the software. The more vulnerabilities, the faster the discovery rate of bugs. Likewise, the more users of the software, the faster the existing vulnerabilities are found (through both formal and adverse discovery).

4.4.2.1 Mapping Vulnerabilities Within Software

Now let *E* stand for the event where a vulnerability is discovered within the times T and T+h for n vulnerabilities in the software:

$$P(E \mid n) = \int_{T}^{T+h} n\alpha e^{-n\alpha t} dt \approx n\alpha e^{-n\alpha T} h$$

Where a vulnerability is discovered between time T and T+h, use Bayes' Theorem to compute the probability that *n* bugs exist in the software:

$$P(n_{vu\ln erabilities} \mid E) = \frac{ne^{-(n\alpha T+\beta)} \frac{\beta^n}{n!}}{\sum_{n=0}^{\infty}\left[ne^{-(n\alpha T+\beta)} \frac{\beta^n}{n!}\right]}$$

Equation (32)

From this :

$$P(n_{vu\ln erabilities} \mid E) = \frac{\frac{\left(\beta e^{-\alpha T}\right)^{n-1}}{(n-1)!}}{\sum_{n=0}^{\infty}\left[\frac{\left(\beta e^{-\alpha T}\right)^{n-1}}{(n-1)!}\right]}$$

Equation (33)

By summing the denominator, it can be understood that in observing a vulnerability at time T after the release and the decay constant for defect discovery is α, then the conditional distribution for the number of defects remaining is a Poisson distribution with expected number of defects $\beta e^{-\alpha T}$.

Hence:

$$P_{\beta e^{-\alpha T}}(n) = e^{\beta e^{-\alpha T}} \frac{\left(\beta e^{-\alpha T}\right)^n}{n!}$$

Equation (34)

Likewise, there exists a method to calculate for *m* users.

4.4.2.2 Exponential Failure

The reliability function (also called the survival function) represents the probability that a system will survive a specified time *t*. Reliability is expressed as either MTBF (Mean time between failures) or MTTF (mean time to failure). The choice of terms is related to the system being analysed. In the case of system security, it relates to the time that the system can be expected to survive when exposed to attack. This function is hence defined as:

$$R(t) = 1 - F(t)$$

Equation (35)

The function F(t) in EQ 31 is the probability that the system will fail within the time t. As such, this function is the failure distribution function (also called the unreliability function). The randomly distributed expected life of the system t can be represented by a density function, $f(t)$ and thus the reliability function can be expressed as:

$$R(t) = 1 - F(t) = \int_{t}^{\infty} f(t)dt$$

Equation (36)

The time to failure of a system under attack can be expressed as an exponential density function:

$$f(t) = \frac{e^{-t/\theta}}{\theta}$$

Equation (37)

where θ is the mean survival time of the system when in the hostile environment and t is the time of interest (the time that the user wishes to evaluate the survival of the system over). Together, the reliability function, R(t) can be expressed as:

$$R(t) = \int_{t}^{\infty} \frac{e^{-t/\theta}}{\theta} dt = e^{-t/\theta}$$

Equation (38)

The mean (θ) or expected life of the system under hostile conditions can hence be expressed as:

$$R(t) = \int_{t}^{\infty} e^{-t/M} dt = e^{-t\lambda}$$

Equation (39)

Where M is the MTBF of the system or component under test and λ is the instantaneous failure rate (Brémaud, 1981) where mean life and failure rate are related by the formula:

$$\lambda = \frac{1}{\theta}$$

Equation (40)

The failure rate for a specific time interval can also be expressed as:

$$\lambda = \frac{\#Failures}{\sum Operating \quad Hours}$$

Equation (41)

Failure rates are generally expressed in terms of failures per hour, percentage of failures per each 1,000 hours or the rate of failures per million hours. For instance, if a system has a 90-day patch cycle (the total mission time) and the total number of software failures in that time is expected to be (or is later measured to be) six vulnerabilities, it is conceivable to calculate the failure rate per hour as:

$$\lambda = \frac{6}{90 \times 24} = \frac{6}{2{,}160} = 0.002778$$

Equation (42)

In the case of an exponential distribution for the system's mean survival under attack, the MTBF can be defined as:

$$MTBF = \frac{1}{\lambda} = \frac{1}{0.002778} = 360 hours$$

Equation (43)

Hence, it is expected the system will survive 15 days before a vulnerability is discovered. This does not return when a system will be exploited, simply the expected probabilistic time that can be used to project and plan future expenditure.

4.4.3 Modelling System Audit as a Sequential Test with Discovery as a Failure Time Endpoint

Combining hazard models[1] with SIR (Susceptible-Infected-Removed) epidemic modelling (Altmann, 1995) provides a means of calculating the optimal information systems audit strategy. Treating audit as a sequential test allows for the introduction of censoring techniques (Chakrabarty, 2007) that enable the estimation of benefits from divergent audit strategies (Benveniste, 1973). This process can be used to gauge the economic benefits of these strategies in the selection of an optimal audit process designed to maximise the detection of compromised or malware infected hosts.

Computer systems are modelled through periodic audit and monitoring activities. This complicates the standard failure and hazard models that are commonly deployed (Newman et al., 2001). A system that is found to have been compromised by an attacker, infected by malware, or is simply suffering a critical but unexploited vulnerability generally leads to early intervention. This intervention ranges from system patching or reconfiguration to complete rebuilds and decommissioning.

Audits and reviews of computer systems usually follow a prescribed schedule in chronological time. This may be quarterly, annually or per any other timeframe. Further, periodic reviews and analysis of systems in

the form of operational maintenance activities also provide for a potential intervention and discovery of a potential system failure or existing compromise.

Using a combination of industry and organisational recurrence rates that are stipulated from a preceding failure and covariate history as derived from the individual organisation provides a rational foundation for modelling current event data. By denoting the number of incidents[li] within the organisation as $\tilde{N}(t)$ by follow-up time *t* and $N(t)$ as the corresponding observed incidents in $(0,t]$ with regards to absolute continuous event times, the hazard or intensity process $\lambda(t)$ for the intervention time *t* using the covariate data $X(t)$ can be expressed as:

$$\lambda(t) = P\left[d\tilde{N}(t)\right] = 1\left[\tilde{N}(u), 0 \le u < t, X(t)\right] \qquad \text{Equation (44)}$$

Taking the assumption that the administrative and audit staff are not the direct cause of an incident, a point process $(T_1, T_2, T_3, ...)$ will usually be observed for the system[lii] being examined. Due to censoring through the audit process, $N(t)$ can be greater than $\tilde{N}(t)$. Equation (41) has an assumption that only a single incident has occurred, that is, $\tilde{N}$ increments by units. Live systems can and do experience multiple incidents and compromises between detection events. Hence, it is also necessary to model the mean increments in $\tilde{N}$ over time

$$d\Lambda(t) = E\left[\tilde{N}(t) \mid \tilde{N}(u), 0 \le u < t, X(t)\right] \qquad \text{Equation (45)}$$

with the cumulative intensity process Λ.

In the case of a continuous-time process with unit jumps, expressions (44) and (45) can be expressed as

$$\Lambda(t) = \int_0^t \lambda(u)\,du \quad \text{Equation (46)}$$

Independent censorship requires that $C \geq t$ (Hsu et al. 2015). This assumption of independent censorship allows the preceding covariate histories to be incorporated into the model. In defining $Y(t) = 1(0 < t \leq C)$, it is now necessary that

$$E\left[dN(t) \mid N(u), Y(u); 0 \leq u < t, X(t)\right] = Y(t)\Lambda(t) \quad \text{Equation (47)}$$

for all times ($t \leq C$) prior to the audit or review.

4.4.3.1 NHPP, Non-homogeneous Poisson Process

Poison processes have been used to model software (Zhu et al., 2002) and systems failures (Marti, 2008), but these models are too simplistic, and it is necessary to vary the intensity (rate) based on historical and other data in order to create accurate risk models for computer systems (Lin et al., 1997). The non-homogeneous Poisson process (NHPP) can be used to model a Poisson process with a variable intensity. In the special case when $\lambda(t)$ takes a constant value λ, the NHPP is reduced to a homogeneous Poisson process with intensity $\lambda(t) = \lambda$.

In the heterogeneous case, an NHPP with intensity $\lambda(t)$, the increment, $N_t - N_u, 0 \leq u < t$ has a Poisson distribution with an intensity of $\lambda(t) = \int_u^t \lambda(x)\,dx$. Hence the distribution function of the incident discovery can be expressed as:

$$\begin{aligned}\Lambda_u &= 1-P\left(N_{u+t}-N_t=0\right)\\ &= 1-\exp\left(-\int_u^{u+t}\lambda(x)\,dx\right)\\ &= 1-\exp\left(-\int_0^{t}\lambda(u+v)\,dv\right)\end{aligned}$$

The NHPP format is better suited to information systems risk modelling than is the homogeneous Poisson process, as it can incorporate changes that occur over time across the industry.

This can also be modelled as the Poisson process with parameter λ, $\left(N_t^{(\lambda)}, t\geq 0\right)$, is the unique (in law) increasing right continuous process with independent time homogeneous increments. Each $t>0, N_t^{(\lambda)}$ has a Poisson distribution with rate λt. The process $\left(X_t^{(\lambda)}, t\geq 0\right)$ is also stationary with independent time increments.

With $t_0=0$ and $(t_1,...,t_n)\in\mathbb{R}_+^n$, $t_1<t_2<...<t_n$ the r.v.'s $\left[\left(X_{t_1}^{(\lambda)}\right),\left(X_{t_2}^{(\lambda)}-X_{t_1}^{(\lambda)}\right),...,\left(X_{t_n}^{(\lambda)}-X_{t_{n-1}}^{(\lambda)}\right)\right]$ are independent and for each $k=1,...,n,\left(X_{t_n}^{(\lambda)}-X_{t_{n-1}}^{(\lambda)}\right)$ has the same distribution as $\left(X_{t_k-t_{k-1}}^{(\lambda)}\right)$.

The characteristic function of $\frac{1}{\sqrt{\lambda}}\left(X_{t_k-t_{k-1}}^{(\lambda)}\right)$ can be computed for any $\lambda_m\in\mathbb{R}$ as:

$$\begin{aligned}E\left[e^{\frac{i\lambda_m}{\sqrt{\lambda}}\left(X_{t_k-t_{k-1}}^{(\lambda)}\right)}\right] &= e^{\left(-i\lambda^{1/2}\lambda_m(t_k-t_{k-1})-\lambda(t_k-t_{k-1})\right)}\sum_{n=0}^{\infty}\left(\frac{\lambda^n\left(t_k-t_{k-1}\right)^n}{n!}e^{\left(\frac{i\lambda_m n}{\sqrt{\lambda}}\right)}\right)\\ &= e^{\left(-\lambda(t_k-t_{k-1})\left(1-e^{\left(\frac{i\lambda_m}{\sqrt{\lambda}}\right)}\right)-i\lambda_m(t_k-t_{k-1})\sqrt{\lambda}\right)}\end{aligned}$$

as $\lambda \to \infty$ the expression converges to $Exp\left(\frac{-\lambda_m^2}{2}(t_k - t_{k-1})\right)$, which is the characteristic function of a Gaussian variable with variance $\sigma^2 = (t_k - t_{k-1})$.

4.4.3.2 Recurrent Events

In many cases, audit and review processes are limited in scope and may not form a complete report of the historical processes that have occurred on a system (Revuz & Yor, 1999). The audit samples selected systems and does not check neighbouring systems unless a failure is discovered early in the testing. In these instances, the primary interest resides in selected marginalised intensities that condition only on selected parts of the preceding histories. Some marginal intensity rates drop the preceding incident history altogether:

$$d\Lambda_m(t) = E\left[d\tilde{N}(t) \mid X(t)\right] \qquad \text{Equation (48)}$$

A common condition for the identification of Λ_m is that

$$E\left\{dN(t) \mid \left[Y(u); 0 \le u < t\right], X(t)\right\} = T(t)\, d\Lambda_m(t). \qquad \text{Equation (49)}$$

For (EQ 46) to be valid, censoring intensity cannot depend on the preceding incident history for the system $\left[N(u); 0 \le u < t\right]$. The process of randomly selecting systems to audit makes it unlikely that particularly problematic systems will be re-audited on all occasions. This would include the exclusion of targeting client systems that have been compromised several times in the past or which have suffered more than one incident in recent history. The result is that covariates that are

functions of $\left[\tilde{N}(u); 0 \le u < t\right]$ will also have to be excluded from the conditioning event. Here

$$E\left[d\tilde{N}(u) \mid X(u)\right] = E\left[d\tilde{N}(u) \mid X(t)\right], \qquad \forall \quad t \ge u.$$

Equation (50)

When this occurs

$$\begin{aligned} E\left[d\tilde{N}(u) \mid X(u)\right] &= \int_0^t E\left[d\tilde{N}(u) \mid X(t)\right] \\ &= \int_0^t E\left[d\tilde{N}(u) \mid X(u)\right] \\ &= \Lambda_m(t) \end{aligned}$$

$\Lambda_m(t)$ models the expected number of incidents that have occurred in the system over $(0,t]$ as a function of $X(t)$.

4.4.3.3 Cox Intensity Models

Using a Cox-type model

$$d\Lambda(t) = d\Lambda_0(t) e^{\left[Z(t)'\beta\right]},$$

Equation (51)

with $Z(t)' = \left[Z_1(t), ..., Z_p(t)\right]$ having been created using functions of $X(t)$ and $\left[N(u); 0 \le u < t\right]$, inference differs little to univariate failure time data. The log-partial likelihood function, score statistic and the integral notation for the information matrix may be written respectively as:

$$\ell(\beta) = \log L(\beta) = \sum_{i=1}^{n}\left[\int_0^{\infty}\left\{Z_i(t)'\beta - \log\left[S^{(0)}(\beta, t)\right]\right\} dN_i(t)\right]$$

Equation (52)

$$U(\beta)=\frac{\partial \ell(\beta)}{\partial \beta}=\sum_{i=1}^{n}\left[\int_0^{\infty}\left\{Z_i(t)-\xi(\beta,t)\right\}dN_i(t)\right]$$

Equation (53)

and

$$I(\beta)=\frac{-\partial^2 \ell(\beta)}{\partial \beta \partial \beta'}=\sum_{i=1}^{n}\left[\int_0^{\infty}\left\{V(\beta,t)\right\}dN_i(t)\right]$$

Equation (54)

where,

$$S^{(j)}(\beta,t)=\sum_{i=1}^{n}Y_i(t)Z_i(t)^{\otimes j}\,e^{Z_i(t)'\beta}$$

$$\xi(\beta,t)=\frac{S^{(1)}(\beta,t)}{S^{(0)}(\beta,t)}$$

and

$$V(\beta,t)=\frac{S^{2}(\beta,t)}{S^{0}(\beta,t)}-\xi(\beta,t)^{\otimes 2}.$$

By defining $Z(t)$ in terms of fixed or external time varying covariates, (8) can be further defined by adding additional elements $1\left[N(t^-)=1\right]\gamma_1+1\left[N(t^-)=2\right]\gamma_2+\ldots$ to $Z(t)'\beta$. This would allow the intensity to be altered by a multiplicative factor e^{γ_j} following the jth incident on an individual system when compared against another system without any incidents at the same point in time.

4.4.3.4 SIR (Susceptible-Infected-Removed) Epidemic Modelling of Incidents during Audit

Allowing that a compromised or infected system remains infected for a random amount of time τ, the discovery of an incident by an auditor will be dependent on a combination of the extent of the sample tested during the audit[liii] and the rate at which the incident affects individual hosts. When a host in a system is infected, any neighbouring hosts are attacked and infected at a rate r. The sample size selected in the audit is set as κ and the total number of hosts in the system being audited is defined by K where $\kappa \leq K$. The time between audits (the censor time) is defined by C.

If $C \leq \tau$, an infected or compromised system will be undiscovered and attacking other hosts within the system when the audit occurs. At the end of the time τ, the system is removed as it is either 'dead'—that is, decommissioned and reinstalled or patched against the security vulnerability.

A NSW (Newman et al., 2001) random graph is obtained by investigating the neighbouring systems in the SIR model. From this, the thresholds can be computed.

4.4.3.4.1 Calculations with a constant τ.

First, consider the case where τ is a constant value, and without loss of generality scale time to make a constant. Start with letting p_k be the degree of distribution.

Starting with a single infected host in a system, the probability that j of k neighbouring hosts will be infected is ascertained by:

$$\hat{p}_j = \sum_{k=j}^{\infty} p_k \binom{k}{j} \left(1 - e^{-r}\right)^j e^{-(k-j)r}$$

Equation (55)

Setting μ is the mean of p then the mean of $\hat{p}$ is $\hat{\mu} = \mu\left(1 - e^{-r}\right)$

With the network constructed as an NSW random graph, systems that are compromised in the first and subsequent iterations will each have k neighbours. The value k includes subsequently compromised machines and the host that compromised the existing system. The probability associated with these neighbouring hosts is given by:

$$q_k = \frac{(k+1)\, p_{k+1}}{\mu} \quad \forall \quad k \geq 0$$

Equation (56)

This allows us to calculate the probability that j neighbouring hosts also become infected:

$$\hat{q}_j = \sum_{k=j}^{\infty} q_k \binom{k}{j} \left(1 - e^{-r}\right)^j e^{-(k-j)r} .$$

Equation (57)

Setting ν to represent the mean of q, leads to the mean of $\hat{q}$,

$\hat{\nu} = \nu\left(1 - e^{-r}\right)$.

From this, it is not too difficult to see that for the attack or malware to propagate and infect other systems, it is necessary to have the condition where

$$\nu\left(1 - e^{-r}\right).$$

Equation (58)

Using this condition allows for the calculation of the probability that a particular attack or type of malware will result in an outbreak.[liv] Setting $T = 1 - e^{-r}$ (Newman, et al., 2001) [lv] returns:

$$\begin{aligned} \hat{G}_0(z) = & \sum_{k=0}^{\infty}\sum_{j=0}^{\infty} p_k \binom{k}{j} T^j (1-T)^{k-j} z^j \\ = & \sum_{k=0}^{\infty} p_k \sum_{j=0}^{\infty} \binom{k}{j} (Tz)^j (1-T)^{k-j} \\ = & G_0\left(Tz + (1-T)\right). \end{aligned}$$

Equation (59)

Similarly, it is simple to prove that $\hat{G}_1(z) = G_1\left(Tz + (1-T)\right)$.

As such, the probability that an incident behaves as an epidemic is $1 - \hat{G}_0(\xi)$ where $\hat{G}_1(\xi) = \xi$ is the smallest fixed point in $[0,1]$.

4.4.3.4.2 Calculations with a variable τ .

Next, it is necessary to consider the effects of a variable or random value of τ . This is the probability that a compromised system causes a compromise in its neighbouring system and is:

$$T = 1 - \int_0^{\infty} dt P(\tau = t) e^{-rt}$$

Equation (60)

Again, T is the transmissibility factor.

Newman et al. (2001) asserted that the infection of neighbours is statistically independent. This does not apply to malware, but it offers a good approximation. Interactions in systems and the ability of software to rescan the same systems carry a degree of dependence. Here the time to

compromise may be modelled exponentially with mean λ, such that $P(\tau = t) = e^{-\lambda t}$.

From this it can be shown that the probability of a host not being compromised is:

$$1 - T = \int_0^\infty e^{-\lambda t} e^{-rt} dt = \int_0^\infty e^{-(\lambda + r)t} dt$$
$$= \frac{1}{(\lambda + r)} \quad . \qquad \text{Equation (61)}$$

Likewise, the probability that n hosts in a system are not compromised is:

$$1 - T = \int_0^\infty e^{-\lambda t} e^{-nrt} dt = \int_0^\infty e^{-(\lambda + nr)t} dt$$
$$= \frac{1}{(\lambda + nr)} \qquad \text{Equation (62)}$$

For cases where τ is not constant, Jensen's inequality (Dempster et al., 1977) implies that, for neighbouring hosts, the probability of escaping compromise is positively correlated as:

$$E\left(e^{-nrt}\right) > \left(Ee^{-rt}\right)^n \quad . \qquad \text{Equation (63)}$$

It is now viable to compute the expected number of systems that will be compromised by substituting r_k for p_k and q_k, which yields,

$$\hat{G}(z) = \int_0^\infty P(\tau = 1) dt \sum_{j=0}^{\infty} z^j \sum_{j=0}^{\infty} r_k \binom{k}{j} \left(1 - e^{-rt}\right)^j e^{-r(k-j)t} \qquad \text{Equation (64)}$$

if $r_0 + r_1 > 1$, G is strictly convex as $\hat{\nu} = \nu T$, $\tilde{G}_i = G_i\left(1 + (z-1)T\right)$.

4.4.3.5 Applications to Audit and Review

The first case where $C > \tau$ has a host in the system being discovered as having been infected or compromised before the audit, the rate of infection r determines the chance of other systems being uncovered during an audit. If the time between audits exceeds the time to compromise, the first host is insufficient for the incident to spread (i.e. $\tau \leq C, C < r$); then only the initial host will have been compromised and this will be known prior to the audit.

If $C \leq \tau$, a compromised host in the system will be undiscovered and will be attacking other hosts within the system when the audit occurs. At the end of the time τ, the system is found. As such, if $(C + c) \leq \tau$ (where c is the average time taken to conduct an audit), a compromised host is discovered during the audit through a process independent of the audit.

The alternative scenario and that which is of most interest is where $(C + c) < \tau$. In this case, the incident will not be discovered independent of the audit, and the calculation of the probability that an auditor will discover a compromised system during the audit process may be determined, as there exists a time-limited network function which is coupled with a discovery process that is formulated using the Bayesian prior. That is, a risk professional can calculate the probability that all systems are not compromised given a selected audit strategy that finds that none of the audited hosts have been compromised.

4.4.3.6 False Negatives in an Audit

False negatives result (Jacod, 1975) in an audit where an incorrectly reported result is supplied, noting the organisation as safe when it is not (i.e. no compromise was detected where hosts have been compromised). By letting A represent the condition where the organisation has been

compromised and B represent the positive evidence of a compromise being reported:

$$P\left(A \mid \overline{B}\right) = \frac{P\left(\overline{B} \mid A\right)P\left(A\right)}{P\left(\overline{B} \mid A\right)P\left(A\right) + P\left(\overline{B} \mid \overline{A}\right)P\left(\overline{A}\right)}$$

Equation (65)

Here it is practicable to model the actual rate of compromise in the system, $P\left(A\right)$. Given a network compromise model (EQ 65), it is realistic to substitute the censored time:

$$\hat{G}\left(z\right) = \int_0^C P\left(\tau \le C\right)dt \sum_{j=0}^{\infty} z^j \sum_{j=0}^{\infty} r_k \binom{k}{j} \left(1 - e^{-rt}\right)^j e^{-r(k-j)t}$$

Equation (66)

From this, the results show that the probability of a host not being compromised in the censor time is:

$$\begin{aligned} P\left(\overline{A}\right) = 1 - T &= \int_0^C e^{-\lambda t} e^{-rt} dt = \int_0^C e^{-(\lambda + r)t} dt \\ &= \frac{e^{-(\lambda + r)t}}{-\left(\lambda + r\right)}\Bigg|_0^C = \frac{e^{-(\lambda + r)C}}{-\left(\lambda + r\right)} + \frac{1}{\left(\lambda + r\right)} \\ &= \frac{1 - e^{-(\lambda + r)C}}{\left(\lambda + r\right)} \end{aligned}$$

Equation (67)

Equation (EQ 67) also derives the probability of any single host being compromised between the audits:

$$P\left(A\right) = 1 - \frac{1 - e^{-(\lambda + r)C}}{\left(\lambda + r\right)}$$

Equation (68)

Depending on whether τ is constant or varies, the process can calculate the expected number of hosts that will be compromised in the period between audits as a fixed or variable function. In either event, each calculation is an exercise in Bayesian estimation of the type where a random sample is selected and the defects or failures are analysed:

$$f(x) = \binom{n}{x} p^x q^{n-x}$$

Equation (69)

This binomial distribution is simplified in the case of a false negative (no failures or x=0 from a sample of n hosts):

$$f(x) = \binom{n}{0} p^x q^{n-x} = p^x q^{n-x}$$

Equation (70)

Based on the types of systems, the audit periods can be selected to create an economically optimal choice. In this, using an equation that calculates the expected number of compromised or infected hosts within a censor time allows for the selection of either a fixed audit schedule, C = Constant, or vary C over the course of the system life to maximise the detection, C=C(t).

In conducting this exercise, the cost of the audit and differences that occur would also need to be modelled. The required effort for an audit of 10 hosts in a one-month period is not necessarily linearly related to the audit of 60 hosts in a six-month period. In each case, the individual constraints faced by the selected organisation also must be incorporated.

4.4.4 Automating the Process

The main advantage to a systems engineering approach is the ease with which it can be automated. The various inputs and formula noted

throughout this paper can become inputs into a neural network algorithm (Figure 21). Equation (70) could be modelled in three layers (Figure 22).

Here, an input layer with one neuron for each input (system or application) could be used to map for IP options, malware and buffer overflow conditions, selected attacks, and so on. The system of perceptrons would be processed using a hidden neuron layer in which each neuron represents combinations of inputs and calculates a response based on current data coupled with expected future data, a priori data and external systems data. Data processed at this level would feed into an output layer. The result of the neural network would supply the output as an economic risk function.

In this way, a risk function can be created that not only calculates data based on existing and known variables (He et al., 1992), but also updates automatically using external sources and trends. Many external sources have become available in recent years that provide external trending and correlation points. Unfortunately, most of these services use clipped data[lvi] as the determination of an attack, which is generally unclear and takes time to diagnose where otherwise useful data is lost.

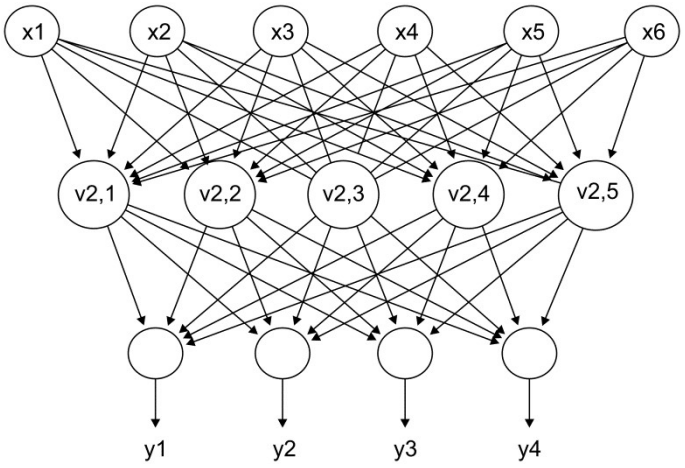

Figure 21. A depiction of a multi-layer layer topology neural network.

Multi-layer layer topology neural networks can be used to accept data from risk models and automatically update the risk profile of an organisation. In modelling risk, each application and system can be modelled using a perceptron.

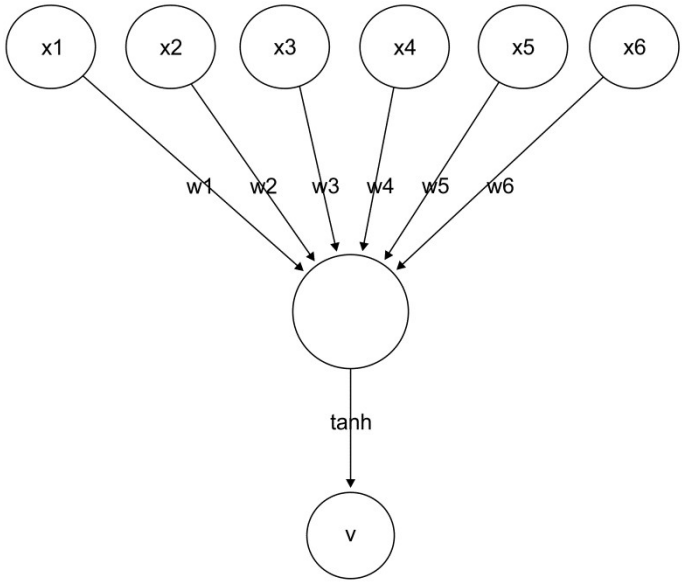

Figure 22. Inputs being fed into a perceptron.

The perceptron is the computational workhorse in this system (Ni et al. 2010). It is reasonable to model the selected risk factors for the system and calculate a base risk that is trained and updated over time. The data from multiple organisations can be fed into a central system (Kay, 1977) that can be distributed to all users. This could be integrated and sold as a product enhancement by existing vendors or independent third parties could maintain external datasets.

$$v_{i,j} = f(\sum_{k=0}^{n} w_{i,j,k} \cdot x_k \ )$$

Equation (71)

Equation 72 defines the input variables as:

- $x_1 \ldots x_n$ are the inputs of the neuron,
- $w_{i,j,0} \ldots w_{i,j,n}$ are the weights,
- f is a non-linear activation function,
- hyperbolic tangent (tanh),
- $v_{i,j}$ is the output of the neuron.

A large vendor such as Microsoft could create an implementation model. In place of offering stale recommended security settings,[lvii] the risk application could automatically collect data from user systems on patch levels and group policy configurations and utilise these to calculate and report on an estimated level of risk and an expected survival time for the system in a number of different scenarios.[lviii]

The training of the network would require the determination of the correct weights for each neuron. This is possible in selected systems, but a far larger effort would be required to enable this process for more generalised deployment. The data needed for such an effort already exists in projects such as DShield, the Honeynet Project and in many similar endeavours. The question is whether there truly exists a will as a community to move from an art to a science.

4.5. Who pays for a security violation? An assessment into the cost of lax security, negligence and risk, a glance into the looking glass

As Carroll (1871) noted, people too often give little thought to the economic allocation of funds to mitigate risk.

> "I see you're admiring my little box," the Knight said in a friendly tone. "It's my own invention—to keep clothes and sandwiches in. You see I carry it upside down, so that the rain can't get in."

> “But the things can get out,” Alice gently remarked. “Do you know that the lid’s open?"
>
> “I didn't know it,” the Knight said, a shade of vexation passing over his face. “Then all the things must have fallen out! And the box is no use without them.”
>
> “I was wondering what the mouse-trap was for,” said Alice. “It isn't very likely there would be any mice on the horse’s back.”
>
> “Not very likely, perhaps,” said the Knight, “but, if they do come, I don’t choose to have them running all about.”
>
> “You see,” he went on after a pause, “it’s as well to be provided for everything.”
>
> “It's too ridiculous!” cried Alice, losing all her patience this time. (p. 181–184)

Just as Alice found risk controls designed to catch mice on horseback ridiculous, many have railed against the controls and processes that provide compliance while being sold as security. Society has created a compliance structure that lets the real controls fall out of the box whilst seeking black swans (Taleb, 2010) against which one needs to defend, taking funds from where they are needed and redirecting them to questionable uses.

An empirical study of data collected from 2,361 information systems audits in the period 1998–2010 (by the author) supports this argument. These audit reports were collected from 894 Australian and US organisations in the finance, gaming, media, FMCG, and mining sectors as well as both federal and state governments. Reports from chartered audit firms, security companies and internal audit contractors were included. The composition of Australian organisations varies greatly. All US organisations consist of medium or larger listed companies with requirements (Bender, 2002) under the Sarbanes-Oxley Act (Sect 3.2 & 404). The audit reports from Australian organisations include PCI-DSS, APRA, BASELII, AML-CTF and those required for listed company

financial reporting. This study incorporated the financial data for 451 of the organisations. An analysis of 210 incidents that resulted in a compromise provides additional support for this hypothesis.

Often, software vendors are blamed for the lack of security in systems, but it is rare to see the auditors and testers called to account. I propose applying concepts of negligence and tort-based responsibility against the inattentive auditor. This would make the auditor liable for the errors and failures with a comparative liability scheme. This would require a radical rethinking of how to go about implementing and monitoring information security and risk (Kunreuther & Heal 2005). In place of testing common checklist items such as password change policy and determining the existence of controls,[lix] a regime of validating the effectiveness and calculating the survivability of the system is proposed.

4.5.1 Misaligned Incentives: Audit and the failure to determine risk

The existing audit industry provides compliance services under the guise of security. Even though these services provide little if any increase in security (Hind, 2004), consumers continue to purchase them (Arora, Krishnan, Telang, & Yang, 2004; Arora, Nandkumar, & Telang, 2006a; Arora et al., 2004). In addition, these services are extremely inelastic for large organisations.[lx] There are several reasons for this. First, government[lxi] or commercial groups (e.g. PCI-DSS) mandate many compliance regimes. Next, negligence rules and the governance functions of companies require that boards and senior management act to protect the value of the company (Grembergen, 2004). Unfortunately, this also means using reports that demonstrate compliance from audit companies in place of a real effort to ensure that data protection occurs (Varian, 2004a).

The courts generally seem willing to apply conventional fault-based tort principles to the behaviour of companies and other organisations (Sugarman, 1996). Not acting to correct a computer system vulnerability can give rise to an action in negligence if another party suffers loss or damage as the result of a cyber-attack or employee fraud. This is, of course, much of the incentive behind auditing in the present context. A favourable audit report can demonstrate good governance to the courts that are unlikely to validate the failures of the audit and compliance process. A combination of proximity[lxii] and reasonable foreseeability (that is, the question of whether there exists a positive duty on a party to act so as to prevent harm or economic loss to others) is evident in the cyber world (Ozment & Schechter, 2006). Problems can thus arise where the organisation remains insecure despite a clean audit report.

To many organisations, the current standards of corporate governance for IT systems pose a problem due to the large number of competing standards. With many of these standards contradicting others, compliance can limit the ability of an organisation to secure its systems; the simple answer for many organisations is to prefer compliance to security. This occurs as a compliance failure and is a greater perceived risk than a security failure. These standards maintain a minimum set of analogous requirements that few companies presently meet. Most of these standards, such as the PCI-DSS[lxiii] and COBIT, impose a requirement to monitor systems. COBIT control ME2 (Monitor and Evaluate Internal Controls) is measured through recording the "number of major internal control breaches".

At minimum, an organisation needs to maintain a sufficiently rigorous monitoring regime to meet these standards. Perversely, the incentive is for the organisation to fail in this control. As auditors using the COBIT framework can determine compliance and extrapolate an apparent

security state through such controls, it is in the organisation's best interest to ensure that no "internal control breaches" are recorded. This can be probabilistically achieved through a set of controls that limits the risk to the organisation and increases its security, or the organisation can implement monitoring controls that are unlikely to find and report on the breach (Marti, 2008).

In a review of 1,878 audit and risk reports conducted on Australian firms by the top eight international audit and accounting firms, 29.8% of tests evaluated the effectiveness of the control process. Of these 560 reports, 78% of the controls tested where confirmed through the assurance of the organisation under audit. A mere 6.5% of systems received validation at any level at all. Of these, the process rarely tested for effectiveness, but instead tested that the controls met the documented process. (Auditing in the US and UK follow a similar practice [Turnbull, S., 2004] as in Australia.)

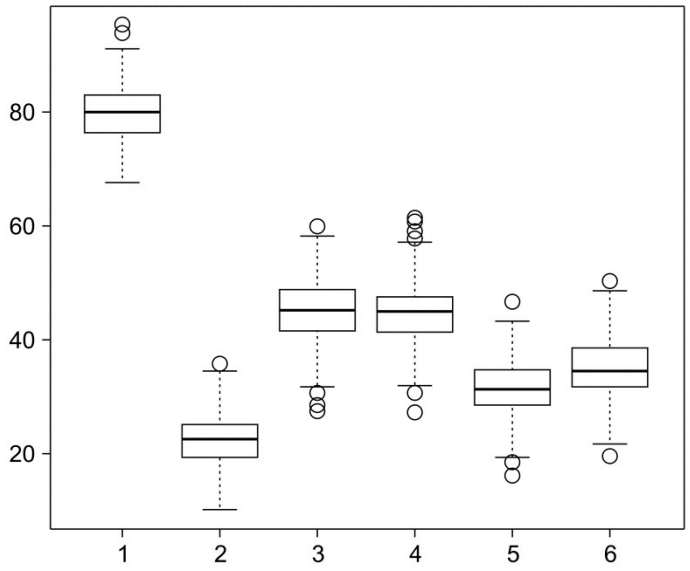

Figure 23. Misaligned incentives and a lack of accuracy delivered to the auditor (%).

Installation guidelines provided by the Centre for Internet Security (CIS)[lxiv] openly provide system benchmarks and scoring tools that contain the "consensus minimum due care security configuration recommendations" for the most widely deployed operating systems and applications in use. The baseline templates will not themselves stop a determined attacker, but can demonstrate minimum due care and diligence. Only 32 of 542 organisations analysed in this chapter deploy this form of implementation standards.

Information systems employees within an organisation also have a misaligned set of incentives (Halderman, 2010). A large component of any audit involves discussions with the relevant employees and

management at the examined organisation. The term auditor is rooted in the act of listening, and its etymology means ‘one who listens’. These same employees’ interests are generally aligned with the audit results. For instance, in 1,325 audits directly involving firewalls, 798 (60.2%) of these audits involved direct interviews with firewall administrators who either had bonuses tied to the outcome of the audit or whose employment was in some manner conditional on the outcome of the audit. Similarly, in 6,541 system audits conducted as a sub-component of the complete audits, 878 administrators from 1,325 (66.3%) had similar constraints.

The consequence of these misaligned incentives is obvious: misinformation. Figure 23 displays the results of the audit when the employee has incentives and knowledge or neither.

Further analysis associated with the assignment of a new auditor followed. The differences between the audits of a known tested system and of a system excluded from testing were statistically significant at the $\alpha = 95\%$ level. At this level, there is a confidence interval of (77, 83) with a corresponding confidence interval of (20, 26) when the employee has incentives and knows that the statements they offer will be tested. The results for an audited employee with incentives whose assertions have been validated (42, 48) when a new auditor is assigned do not differ significantly[lxv] from the employee with no incentives (42, 49).

Employees have few incentives to give information to auditors when they do not expect corroboration of their statements. This process requires validation through testing. Unfortunately, most audits follow the philosophy of auditors being “watchdogs and not bloodhounds”, encouraging auditors to accept the word of the employee without testing. This is also a more profitable option for the audit firm that makes over

300%[lxvi] of the returns of testing with confirmation compared to simply trusting the assertions of the organisation's employees.

Hence, it is prudent to validate whenever possible and have a significantly large sample of systems tested such that the organisation expects to be tested. The incentives of the auditor and employee need to align to guarantee the process will be functional.

4.5.1.1 Patching and Validation: what if the king's men know the egg is to fall?

Patching is a test for compliance. Auditors assert that this compliance test aligns with good security practice. A correctly patched system is likely to be more secure. Yet the question is, what signifies "correctly patched" and "have the patches been applied correctly?" Audits generally test for the application of patches. However, this is generally limited to testing the existence of operating systems (e.g. Windows 2003 server) with all required patches applied. Application patches are another matter.

Tests of the patching processes for Windows servers, clients, applications, routers, switches and firewalls are reported in Table 7. The 95% confidence intervals for patching times for each of these systems have been recorded. These results are displayed as boxplots in Figure 24. The patch date is determined as the difference in time between when the software vendor has released the patch to the installation of the patch on the system. In a few instances, this result is statistically censored due to the lack of patching. This can take place where the system is installed and left running without the application of updates. In this case, the difference between the installation date of the device and the date of the patch or update that should be applied is used to determine the interval. This situation was found to be most common in network equipment (with

several routers and switches never having been patched or updated) as well as with selected examples of user application software.

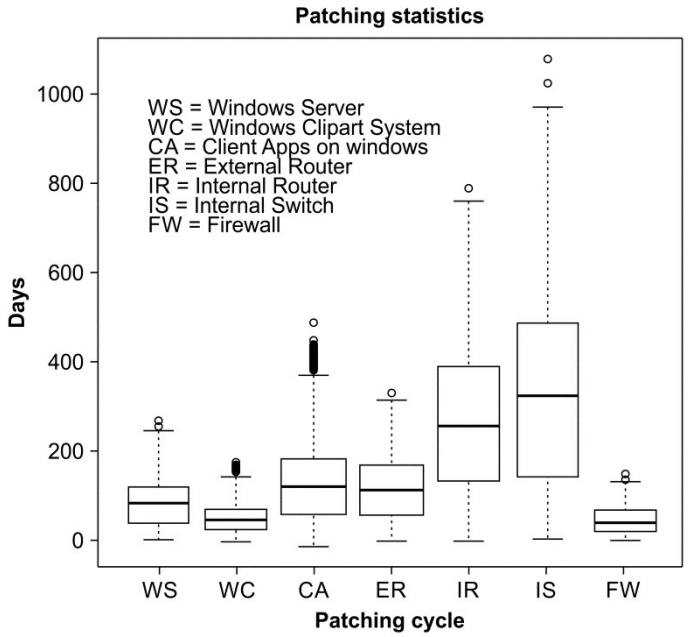

Figure 24. Patching, just enough to be compliant, too little to be secure.

A further analysis of prior audit reports was conducted to note how many of these had included patch levels for each of the various hosts and systems deployed at the audited client. Nearly all audit reports note the inclusion or exclusion of operating system patches (98.4% and 96.6% for server and client systems respectively). Most these reports included no testing of the network devices and few tests of the application software in use by a client. Consequently, there is little incentive for the organisation under audit to maintain critical systems. Network switches were the least analysed device. Consequently, the mean time between patching on these devices was recorded at 341.2 days. It was also uncommon for organisations to have a policy requiring the patching of network devices. Whereas most organisations have policies in place for the patching of servers (with a range of 55.5 to 87.9 days at a 95% CI) and Client

operating systems (with a range of 29.6 to 49.4 days at a 95% CI), most organisations did not have a policy for the patching of network devices.

With a few exceptions, operating system patches for client systems and firewalls are applied and tested within 60 days. The patching rates for network equipment vary significantly.

Again, the incentives to ensure compliance result in insecure systems. The audit process checked policy statements against a sample of systems, but did nothing to validate those systems not included in the policy. The result is an overwhelming focus on selected systems that are incorporated within a checklist at the expense of excluding many essential systems.

The patching of client applications was problematic, with a mean of 125.2 days between patching of these applications and a 95% confidence interval of (58.1, 181.8) days. This varied widely not only across hosts and organisations, but also within the same host. Only 2.18% of hosts patched at least 95% of applications within 120 days. The development systems analysed exhibited the worst results. Compilers and IDE (integrated development software) were patched at a rate of between (82.0, 217.3) days. These systems were also generally not included within the audit report, indicating that there is little incentive for the organisation to ensure that they are maintained sufficiently.

Table 7 Patching Analysis of Audited Systems

	No. Analysed	**95% Confidence Interval of days between patching (Mean)**	**Average Policy Patch time (CI)**	**% Prior Reports noting patching**
Windows Server	1571	41.1, 122.4 (86.2)	55.5, 87.9	98.4%
Windows Client	13,951	22.8, 69.3 (48.1)	29.6, 49.4	96.6%
Other Windows	30,290	58.1, 181.8 (125.2)	68.1% NA	18.15%

Applications				
Internet Facing Routers	515	58.2, 164.1 (114.2)	58.1% NA	8.7%
Internal Routers	1,323	129.3, 384.6 (267.8)	73.2 NA	3.99%
Internal Switches	452	139.9, 483.9 (341.2)	87.5 NA	1.2%
Firewalls	1,562	21.5, 65.7 (45.4)	24.5, 108.2	70.7%

As with the white knight in Alice's adventure, an audit of a system is often of little use if the information has already left the box. Unfortunately, audits and compliance reviews do not test network extrusion. They do not test if data is leaving the organisation or if it has already left. The testing of extrusion data and an analysis of network traffic leaving the organisation was included in the scope of only 1.55% of the audits reviewed.

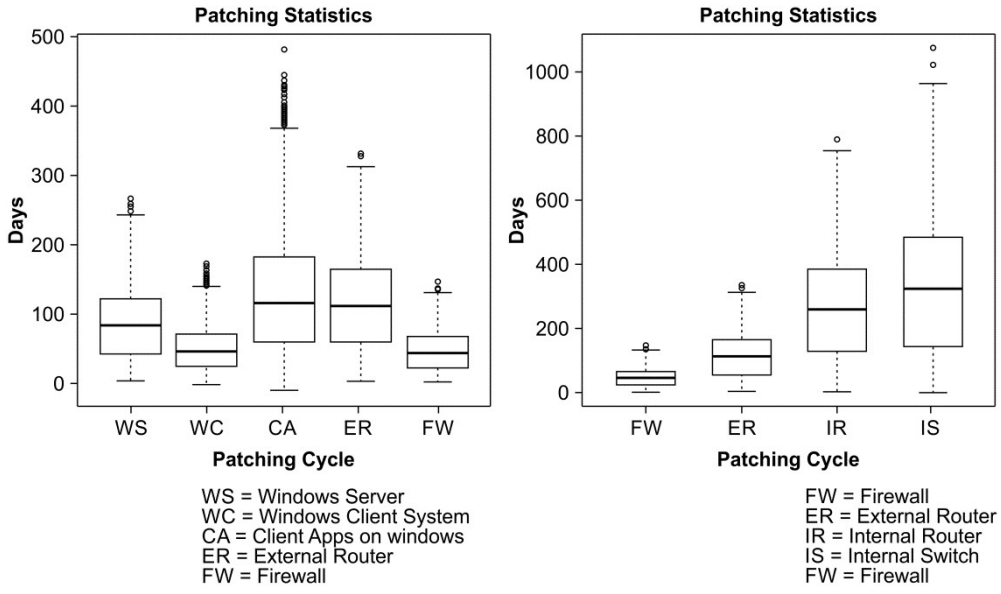

Figure 25. Patching the primary systems.

Validation of data was only marginally better. In the sample of 1,562 firewall systems and the associated reports that have been reviewed, only 4.55% of reports conducted a validation of the firewall through a process involving a test of the device using network packets. The remaining 95.45% of audits reviewed the firewall policy and conducted no tests that would determine if the policy had been deployed. Of these systems, 3.02% had been discovered to be installed and configured incorrectly. The result of this was that the policy that was reviewed in the audit process could not be applied and hence the firewalls had been running with a default open policy.

The incentives of an information systems audit in the existing environment are not structured to ensuring the optimum minimisation of risk and levels of security. The auditor is rewarded for finding an 'acceptable' level of control flaws that ensure an appearance of diligence, but not too many. This process is also aligned to compliance where governance is viewed as a function of meeting "consensus guidelines" for controls that are simple and inexpensive to validate (such as password length and change times). Unfortunately, these controls and compliance checks do little to ensure that systems are secured. Worse, more expenditure on compliance reduces the funds available to be expended on security.

4.5.2 Negligence: who is at fault when a breach occurs?

This study demonstrates that existing rules are inadequate to ensure the accurate outcome is achieved during an audit and that the creation of liability and insurance rules for the auditor could help reorient the function of audit from one of mere compliance to one with a more practical focus on risk and security. The legal rules of negligence (aka Tort) were devised from the economic objective of optimality. An

injuring party X_1 is negligent if X_1 spends less than the optimal amount X_1^{Ω} in both preventing and mitigating the risk of damage occurring against X_2. The damaged party X_2 is negligent where the amount spent in ensuring it is not damaged (does not suffer from the risk posed by the action of others such as X_1) is less than the optimal amount, X_2^{Ω}. As such, negligence occurs where $x_1 < X_1^{\Omega}$ or $x_2 \leq X_2^{\Omega}$.

The optimal values, X^{Ω} and X_2^{Ω} occur where the returns on investment, x_1 and x_2, equate to at least \$1. That is, the expenditure on protection is less than the expected losses $x_1 + 1 \leq E_{Damage}(x_1)$. The end goal of any negligence rule is the choice of the optimal level of prevention and mitigation such that equilibria are created with each party (X_1 or X_2) acting optimally.

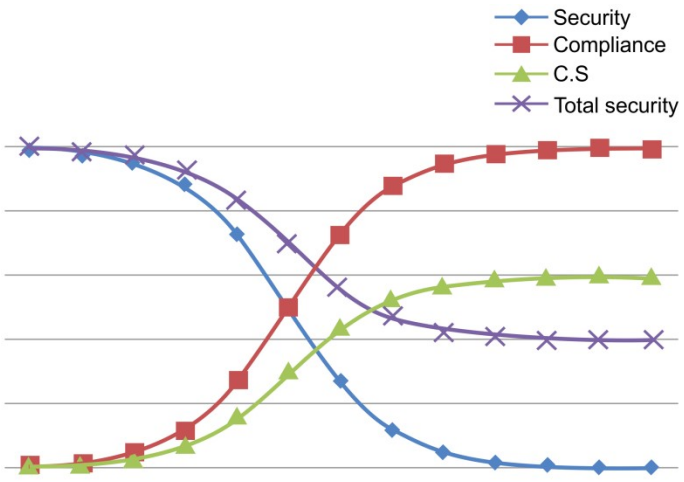

Figure 26. Security vs. compliance

The reality is that all parties will either mis-estimate the risk posed in any given situation or be faced with the results of the mis-estimating of others. Security and compliance spending are dependent. As such, the total cost of risk mitigation is a function of both security [$C_{Security}(x)$] and

compliance [$C_{Compliance}(x)$] minus the cost of security where compliance objectives are met. That is, cost [C(x)] is defined as:

$$C_T(x) = C_S(x) + C_C(x) - \left[C_S(x) \cap C_C(x)\right]. \qquad \text{Equation (72)}$$

The optimal expenditure on the prevention and mitigation of risk C^{Ω} is a function of security and not of compliance, thus, $C^{\Omega} \le C_T(x)$ and the condition $C^{\Omega} = C_T(x)$ only occurs when compliance and security goals completely overlap. Unfortunately, all compliance and security efforts incorporate transactional costs. Namely, the cost of ensuring a secure system is also compliant when all the relevant standards and laws cost more than creating a secure system alone.

Figure 26 displays the relationships between security and compliance for an organisation. A given maximum level of security can be obtained with the resources supplied. These resources can be allocated towards either a compliance or security function with the resultant level of security delivered as a function of the amount allocated to compliance. By assuming all risk funds are allocated directly to either security or compliance (with some security benefits at a level θ where $0 \le \theta < 1$ of Security), one can derive the functional level of security provided from compliance. As such, the level of security (S) delivered from a compliance effort is $S = \theta C$ and the total security of the system delivered is $S_{Tot} = S_C + S = \theta C + S$. As the total expenditure is derived from security + compliance with a dollar of compliance funds being taken directly from security funds,

$$\begin{aligned} S_{Tot} &= \theta C + S = \theta(1 - S) + S = \theta C + 1 - C \\ &= S - \theta S + \theta = S(1 - \theta) + \theta \end{aligned} \qquad \text{Equation (73)}$$

Here θ represents the effectiveness factor of the compliance regime. As demonstrated above, the audit process aligns with compliance with a comparatively small function based on the true assessment of security risk. This misaligned incentive increases the disparity between security and compliance. Negligence is a function of compliance. However, the risk that a given incident will occur is a function of security efforts. All compliance regimes require that the system be tested in accordance with the compliance rules. This has a cost. As it is possible to build a system that is optimally secure but which has not been tested against a compliance regime, it is possible to have no compliance costs with optimal security. This will of course be a non-compliant system and, under existing negligence- and compliance-based rules, a negligent one.

4.5.2.1 Comparative Negligence

Under a comparative negligence rule, the relative negligence is compared with both parties' damage and culpability. As the level of negligence associated with the at-fault party increases, the percentage of damage awarded also increases. This would require that each party's performance be measured in comparison with an optimal level of performance. Where one party is not negligent, the negligent party remains liable for 100% of damages.

This situation is complicated if both parties are negligent. As an example, if comparative negligence was applied to the software industry, the software coding practices of the vendor would be compared to an optimal level with the vendor being found to be negligent where the coding and development practices that are used are sub-optimal. This would be where the last dollar sent on mitigating the vulnerability saves a dollar in loss through risk mitigation to the vendor. That is, any further expenditure on securing the software would result in lost profit. As

Telang and Wattal (2004) demonstrated, this is an unlikely situation. As the vendor, can be expected to suffer a capital loss of around 0.63% of the company's market value for each vulnerability, the probability of a firm remaining in business and neglecting to secure their code to the optimal level is low in the long term.

On the other hand, the user of the software would be negligent if they failed to install the software in an optimal manner such as would be required to minimise the risk of a security breach when measured against the cost of securing the system. The result of such a scenario would be the company asserting it had adequate security in place to provide an optimal (though not perfect) level of security while the vendor would assert that it had found the optimal level of code flaws and has had its programmers correct them sufficiently. Such a disagreement would presumably result in excessive losses for each party and a strong possibility of extraordinary losses for the vendor (which would then pass these costs to the consumer).

This matter is further compounded as the vendor would rightly claim interactions from other vendors. These interactions can and often do lead to cascade failures in systems. The vendor would be able to share damages with other vendors that are then also assumed liable. Any single system can have hundreds of applications for many tens (or more) of vendors. The level of interactions is at best unpredictable. In addition, the vendor is in a state of incomplete information.

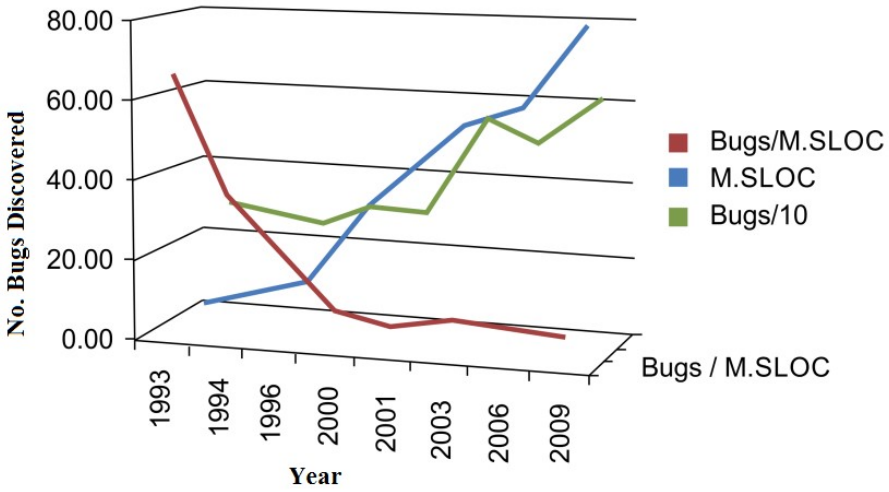

Figure 27. Bugs as a function of time and code.

The reality is that the vendor and external parties can analyse published software vulnerabilities equally and that there are more external users of most software products giving the users more resources than the vendors. For a percentage basis (that is, vulnerabilities / SLOC), the level of vulnerabilities in code has decreased significantly each year for the past decade (Figure 27). Although the number of bugs[lxvii] in any individual software product is increasing, the volume of code is increasing faster. The number of bugs for each line of code has decreased significantly.

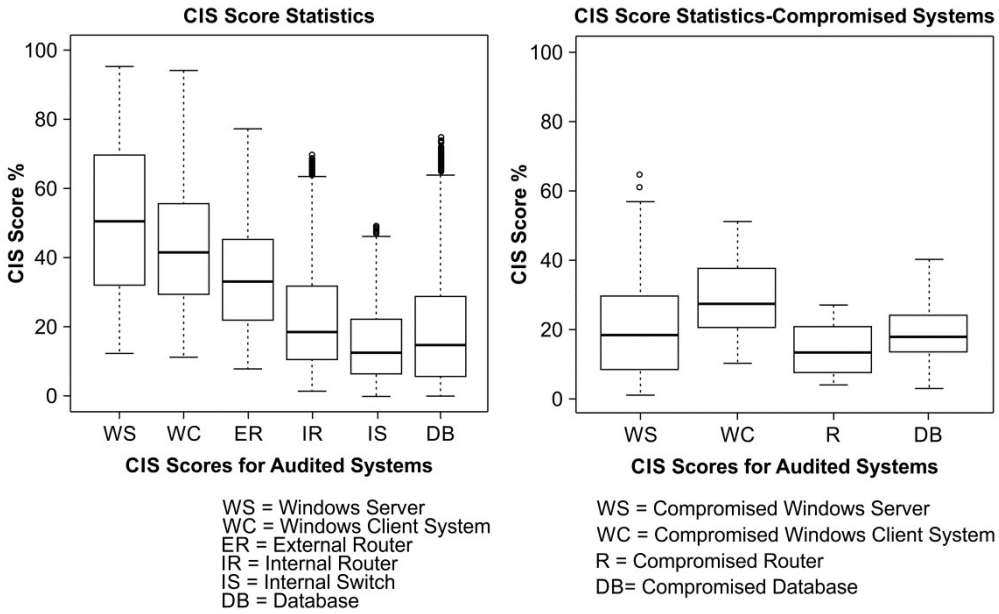

Figure 28. CISecurity.org rating.

Katz & Shapiro (1985) asserted that vendors can easily hide vulnerabilities in service packs rather than releasing them as hotfixes and interim patches. This may have been a factor many years ago, but this is no longer the case (Campbell et al., 2003). It is common practice for penetration testers and malware authors to use service packs to design better attacks (Hofmeyr, et al., 2011). Using the altered code, the attacker can quickly find the areas in code that the service pack has modified. These can be compared with those in the hotfixes to uncover any remaining "hidden" vulnerabilities. With knowledge of the section of vulnerable code that is being patched, an attacker can target the patched section alone. This process will allow the attacker to create an exploit and attack unpatched systems. As most corporate systems have a mean patch time for servers of 86 days and client or user systems 48 days, this provides a large window of opportunity. Service packs also have more avenues to attack than a simple hotfix as they apply to more areas of code.

The user can estimate the remaining number of bugs in code based on the prior coding practices of the vendor and the mean number of errors using Bayes' formula (Kuo & Yang, 1996). Thus, the information held by the consumer is like the vendor (some possessing more knowledge, others less).

Worse, the vendor is unlikely to hold knowledge of the deployment state at the consumer. The vendor can publish installation guidelines and security configuration documents. In addition to the vendor's own recommendations, the user can freely source third party configuration, security and audit guidelines for most common products. In the process of analysing the organisations, the evaluation tool from CISecurity[lxviii] was used on each of the systems. These benchmark audit tools offer an accurate and repeatable audit benchmark. This data is presented in Figure 29 for audited systems (left) and those experiencing incidents (right) that led to a compromise. The CIS scoring data was collected on 1,254 of the 2,361 audits between 2005 and 2010.

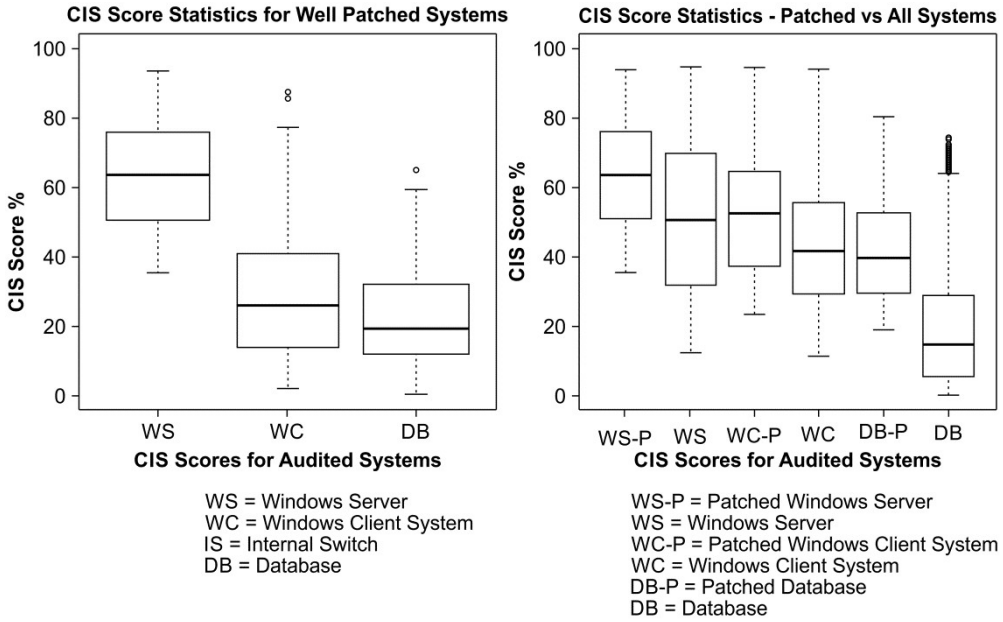

Figure 29. CISecurity.org rating for well-patched systems.

A subsequent analysis was conducted on 153 audit reports where the patching processes at the organisations were noted as being exemplary. The CIS summary figures for these 153 systems are displayed in Figure 29.

Table 8 *Wilcoxon Rank Sum Test* $\alpha = 5\%$.

95% Confidence Interval of CIS Scores, (Mean) %	Base—Compromised	Compromised—Well Patched
Windows Server	W = 1550088, p-value < 2.2e-16	W = 169725, p-value < 2.2e-16
Windows Client	W = 619629, p-value = 2.523e-10	W = 7094, p-value = 4.441e-16
Databases	W = 49219, p-value = 0.659	W = 466, p-value = 0.0007551

A Wilcoxon rank sum test (Table 8) of the data (Table 9) demonstrates that there is a significant difference in the complete population to the compromised systems. The Windows server and client systems that have been compromised are also significantly different. The base rates and compromised results for databases were not statistically significant at the $\alpha = 5\%$ level. No system with a CIS score of greater than 70% in any class was compromised.

Table 9 Analysis of Patched and Compromised Systems

95% Confidence Interval of CIS Scores, (Mean) %	Base	Compromised	Well Patched
Windows Server	31.9,69.5 (50.7)	8.02,29.2 (20.7)	50.9,76.0 (63.5)
Windows Client	29.2,55.4 (43.4)	20.4,37.2 (28.4)	37.4, 52.4 (64.6)
Internet Facing Routers	22.1,45.1 (34.5)	7.24,20.4 (13.6)	NA
Databases	5.48,28.9 (19.2)	13.1,24.1 (19.3)	29.6,52.6 (42.7)

Low expectations of system compromise arise when that system is secured per the recommended security guidelines. An analysis of the compromised systems against the SANS consensus audit guidelines[lxix] was planned for this thesis. Unfortunately, no system that met at least 50% of the control guidelines was found in the compromised list. None of the organisations with compromised systems had implemented systems to monitor and record these controls. This requires further research.

This section does not demonstrate that incompetent or negligent software vendors do not exist, but rather that these do not represent the norm. The end-user organisations can also deploy additional controls on their existing systems for little cost. Most of these controls are available but are not deployed. The low control scores (Figure 29, left) returned from the audits together with the low patching rates of network equipment (Table 7) demonstrate that organisations focus on meeting compliance goals over security. In many cases, management within those organisations does not recognise a distinction. It is held that compliance and meeting audit targets is a reliable indicator of effective governance and resilient security.

Of the 515 Internet facing routers included in this study, only four had BOGON filters and ACLs (access control lists) were only applied to 51 devices. Most organisations trusted the firewall and did not implement ACLs even for protecting access to trusted administrative features of the router. The use of ACLs could have helped stop the BGP flaws and other router attacks that have occurred in recent times. The use and deployment of ingress and egress filters for unknown networks would have also helped stop many of the compromises included in this study.

No analysis of zero-day vulnerabilities has been incorporated into this paper. The reason for this is simple. Of the 210 compromises analysed,

no zero day vulnerabilities were included. The entire reported compromise history of the 894 organisations included in the study included one previously unreported attack. This occurred against a Cisco router and the patch that would have prevented this was made public over 200 days before the incident.

4.5.2.2 Externalities and auditors

Compliance is easier than security and is also more profitable for the organisation providing the service. The audit company does not pay for the failure to note a security breach, and the cost of security is not borne by them. Instead, the software vendor is blamed for the user's inappropriate actions. These are actions that should have been noted by the auditor and rectified.

The user is often in a state of complete ignorance as to what software they have deployed and how. This is no fault of the software vendors and instead stems from negligence on the part of the consumer. This is one of the aspects of an effective audit, but one that does not occur, as evidenced by the results of this study.

4.5.3 Least cost transactions

Transaction costs exist when measuring risk, and adding a requirement for comparative negligence in software security would introduce monitoring and other enforcement costs that would likely be passed on to the consumer. External monitoring and oversight is necessary in this example as all parties attempt to lower the cost of doing business by avoiding the obligations that have been imposed on them. This includes monitoring the vendor as well as the user. As monitoring is an audit function, the vendor and user will align their actions to the compliance of the audit. This does not create a more secure system, but one that is easier

to check (as checklists tend to lower transactional costs). Even software that has been formally verified can still fail; indeed, the formal verification may not be correct. In this instance, the code may conform perfectly to the formal statement, but fail in its real task.

The least-costly provider of security services is often the consumer. An investment of a few hundred dollars and some time spent configuring the system correctly do more for the totality of software installed on a system than any single software vendor could. Unfortunately, the results of this study demonstrated that most organisations focus on compliance at the expense of security. Spending on black swans is permissible, if there is some basis for spending and it does not significantly reduce the funds available for mitigating more likely issues. This should be the focus of audit.

Incomplete information is not to be confused with imperfect information, in which players do not perfectly observe the actions of other players. The purpose of audit is to minimise the probability of incomplete information being used by management. For this to occur, information needs to be grounded in fact and not a function of simplicity and what other parties do. Most security compromises are a result of inadequate or poorly applied controls.

4.5.4 Conclusion

It would seem costs of normal compliance auditing do not benefit the bottom line financial posture of organisations seeking to be both secure and compliant. An appropriate view would be to seek to be secure in place of appearing secure. This leads to an endless cycle of continual audit satisfying the needs of compliance and the bottom lines of financial firms, but little practical payback. But at what price?

The practice of implementing monitoring controls that do not report on breaches, but which do satisfy the compliance needs of an organisation can cost far more in the long term.

Businesses should demand more thorough audits and results that go beyond simply meeting a compliance checklist. These must include not only patching for all levels of software (both system and applications) as well as the hardware these run on. This failure of audits to be more proactive or diligent in favour of merely acting as a watchdog could prove inauspicious (even negligent) for all parties.

A first step is to re-evaluate the assignment of risk. Compliance at the expense of security in the global economy is a practice that is difficult to overcome, but a challenge that must be met.

4.6. Chapter Conclusion

The purpose of information security according to Wright (2008) is to preserve:

- Confidentiality: Data is only accessed by those with the right to view the data.
- Integrity: Data can be relied upon to be accurate and processed correctly.
- Availability: Data can be accessed when needed.

It is rare to find that the quantification of an externality or the quantitative and qualitative effects on those parties affected by, but who are not directly involved in a transaction, has occurred, even though this calculation forms an integral component of any risk strategy. The costs

(negative) or benefits (positive) that apply to third parties are an oft-overlooked feature of economics and risk calculations. For instance, network externality attributes positive costs to most organisations with little associated costs to themselves. In these calculations, the time-to-market and first-mover advantages are critical components of the overall economic function with security playing both positive and negative roles at all stages of the process.

The processes that can enable the creation and release of actuarially sound threat-risk models that incorporate heterogeneous tendencies in variance across multidimensional determinants while maintaining parsimony already exist in rudimentary form. Extending these though a combination of heteroscedastic predictors (such as GARCH/ARIMA), coupled with non-parametric survival models, will make these tools more robust. The expenditure of further effort in the creation of models where the underlying hazard rate (rather than survival time) is a function of the independent variables (covariates) provides opportunities for the development of quantitative systems that aid in the development of derivative and insurance products designed to spread risk.

Good security practice minimises risk through stopping known events, and at the same time buffers against black swans and other outlier incidents.

The equations presented in this chapter allow organisations to compare the deployed risk strategies against both their own historical data and that of third parties. In this manner, strategy can be formulated to optimise audits and system reviews to detect an incident in the most economical manner. These risk-derived processes could offer significant social benefit if applied prolifically.

Modelling the failure rate of systems and the propagation rate of an attack allows one to calculate an expected number of hosts that are anticipated to have been compromised in the time between an audit given a specified survival function or threat. Past data and comparisons from similar systems (such as survival data from DShield) allow for the modelling of alternative systems where a reported number of events have been reported against those deployed.

Dependence, variation, randomness, and frailty add to the risk toolset of multivariate failure event analysis. Using frailty theory to model information system risk allows one to better predict risk and to more effectively allocate scarce resources through selecting the most economically viable targets to defend as well as choosing the optimal detection strategies. The properties of censoring-handling and frailty modelling have turned multivariate survival analysis into an exceptional tool for the determination of system risk.

Any attack will have several stages, and it is important that a security administrator understands these states to be able to:

1. Mitigate attacks before they cause damage,
2. Log an evidence trail for possible prosecutorial use,
3. Defend against possible attacks against the organisation.

As this chapter demonstrates, it should be feasible to stop all attacks from unskilled attackers and to make it economically impractical for skilled attackers to spend time attacking a system. An understanding of how an attacker thinks is essential to this process but mostly, the use of simple controls that increase the survival time and minimise the economic benefits of hacking will do more to secure a system than many costly alternatives.

As was demonstrated in Wright (2011e), Wright (2011f), and Wright & Zia (2011a), many security issues stem from a misalignment of compliance and security. Unfortunately, legislative mandates for security controls are sometimes ineffective, shifting the focus to compliance with the law itself while failing to bolster the security that such laws are designed to achieve. However, when applied correctly, controls can increase the security of a system by enabling a greater survival time and rate and by minimising the hazard posed to that system.

As demonstrated in Wright (2010b), simple controls that are often overlooked and which frequently come with the system without extra expense or at most, small incremental configuration charges can provide significant increases in survival time. Taken another way, the implementation of the correct set of controls can be both measured and demonstrated to influence the survivability and hence security of the system. Using comparative metrics in place of absolutes, it is possible to measure the effectiveness of a security control. This necessitates comparing the system with and without the control or comparing the survivability of a system with one set of controls against an alternative configuration.

Chapter 5 Human factors in IS risks

5.1. Introduction

This section extends the analysis to the human aspects of security. As was demonstrated previously, the incorrect application of incentives can lead to increased compliance or other goals at the expense of security.

"The Not-so-mythical IDS Man-Month: Or, Brooks and the rule of information security" (Wright, 2010a) offers modelling of incident detection and response processes within organisations. The paper examines the effects of interactions as people are added to teams and demonstrates that just as too many cooks spoil the broth, too many incident handlers cause economic loss and reduced reaction time within an organisation.

In "Using Checklists to Make Better Best" (Wright & Zia, 2011f), it is demonstrated that the more routine a task is, the greater the need for a checklist (especially in urgent situations). The use of standard checklists and flowcharts created by the individual makes for better results even in daily tasks. This chapter presents the results of an experiment into the use of checklists by incident responders and quantifiably demonstrates how basic checklists can improve an organisation's security.

The chapter concludes with three sections that investigate the impact of management and human resources on security. In any discussion of reward management, it is essential to first define what is being rewarded

and why. IT is a changing environment where the traditional "independent loaner" is slowly being supplanted by the team player. Reward systems need to take this into account and treat staff on a varying basis. Management's role is to accomplish production through others. For this reason, management is more comfortable when employees are directed, and committed to achieving the organisations objectives. In this it is essential that management creates the most effective method of developing their IT employees if they are to incentivise behaviour that minimises risk and increases security.

The greatest threat to an organisation's security comes from inside its own walls: staff, ex-staff and consultants are the greatest risk faced by any organisation. Most of the risk is a direct result of inadequate HRM processes and awareness. The rise in IT governance legislation and other requirements has driven organisations to monitor and implement controls over the human resources operations.

Not only does this process make them more effective when implemented appropriately, it helps make the organisation more secure (Hawkins, Yen, & Chou, 2000). The cost of security now leads to savings later.

5.2. What does this mean for ITC Professionals?

As "competency-based approaches to management development are most likely to be useful in large, mechanistic bureaucratic organisations which have clearly delineated roles and functions that are well documented" (Toohey, 1995), information technology professionals may face difficulties in adjusting to this style of control. "Faster and more flexible ways to respond to management development needs may be what is required in the present turbulent management environment" (Toohey, 1995) of IT where change is a daily aspect of the job. IT roles are often comparatively autonomous in nature, requiring a large degree of independence. Bureaucratic systems of control generally leave IT professionals feeling they are being watched too closely, and unless supervisors are given a structure to work from that reflects the objectives of the executive management, their observations may reflect their own biases, rather than the objective performance of employees (Lane, 2004), as they are not trained in behavioural assessment skills.

In performance appraisal, there are conflicting strains and prospects for both employers and employees (Carter, & Jackson, 2000). The ideal approach to performance management is thus an intangible goal that is often only vaguely defined. Lansbury further remarks that "a well-designed system, based on objective performance criteria negotiated between management and employees, and providing for two-way feedback and communication, may achieve worthwhile outcomes" (Lansbury et al., 1995).

According to Stone (2002), the aim of a performance appraisal is to:

1. improve employees' work performance by helping them to realise and use their full potential in carrying out their firm's missions,
2. to provide information to employees and managers for use in making work-related decisions.

Specifically, appraisals may provide legal and formal organisational justification for employment decisions to promote outstanding performers while also removing the marginal and low performers (Williams, 2002). They also serve to train, transfer and discipline others while justifying or denying merit-based salary increases. Finally, they are also the foundation of a legal method to reduce the size of the workforce (Kunreuther et al., 2002).

It has been asserted (Toohey, 1995) that appraisal results are correlated with test results from management studies in order to evaluate the hypothesis that test scores predict job performance. Appraisals also present feedback to employees, which assists them in furthering their own personal and career development goals. This may also present both the employee and management with opportunities to develop and instigate training programs.

Toohey (1995) also notes that the appropriate specifications of performance levels developed from appraisals can help detect "organisational problems by identifying training needs and the knowledge, abilities, skills, and other characteristics to consider in hiring". Appraisals are the commencement of the process, rather than the end, as they provide a basis for distinguishing amongst successful and unproductive employees.

This chapter looks at the effects of interactions as people are added to teams and demonstrates how too many incident handlers may complicate

the resolution of a problem, with inauspicious financial consequences for a company.

5.3. The Not-so-mythical IDS Man-Month (or, Brooks and the Rule of Information Security)

Brooks (1995) analysed the concept of the "man-month" in software engineering, a long-standing idea in coding theory. Here, a "man-month" is defined as the amount of work one person could achieve in a month or a multiple of people could achieve in the correspondingly reduced timeframe. Using similar forms of analysis, one can demonstrate that incident response, intrusion analysis and other complex security tasks form "tasks with complex interrelationships".

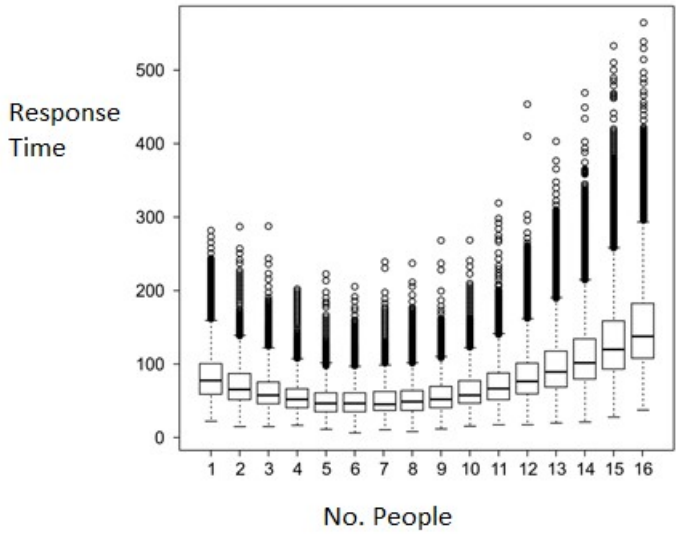

Figure 30. Response time against people.

This research incorporates a small amount of data from a five-year period on incidents affecting 165 companies and other organisations with an Internet presence and some level of security response capability. The

research incorporates an analysis of 423,000 incidents. Costs are based on reported financial figures derived from the actual financial and accounting records for these firms. The data was collected between Australian fiscal years 2003 and 2008.

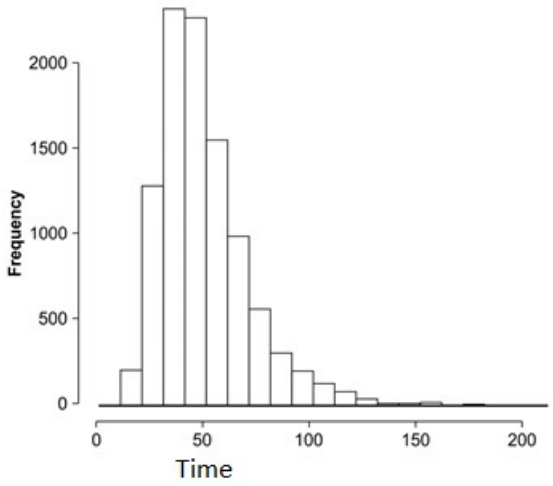

Figure 31. Six-person response times.

Figure 30 lists the individual incident response data for the times (in minutes) against the number of personnel involved in the process (including management) and shows how the data is right-skewed. This is clearly displayed in the histogram of incident response times of six-person teams (Figure 31).

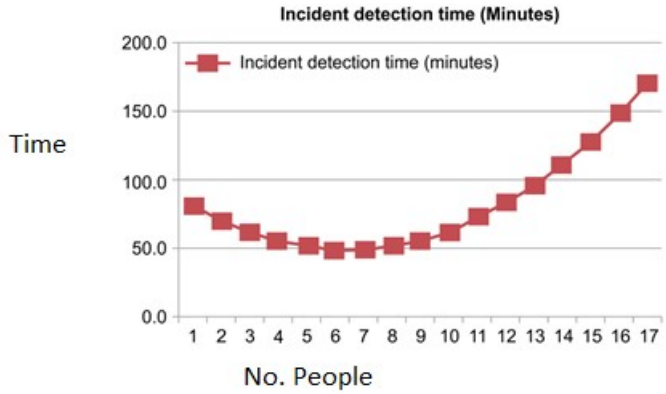

Figure 32. Optimal (mean) detection time and team size.

Figures 32 and 33 incorporate a simplification of the results of Figure 30. The data is overlaid in Figure 34. This plot is summarised with only the mean (average) values being reported.

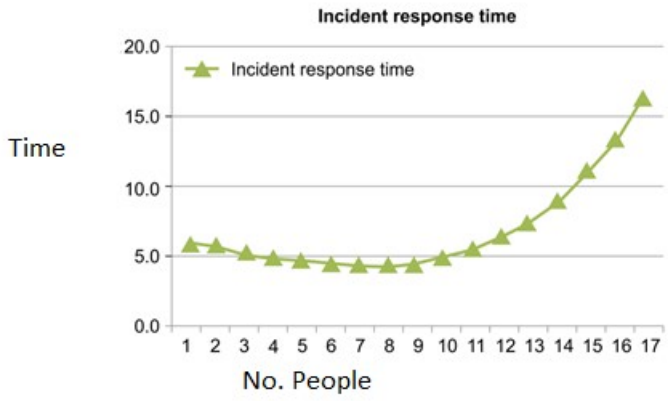

Figure 33. Mean incident response times.

This exemplifies Brooks' (1975) supposition that in tasks with complex interrelations, "the added effort of communicating may fully counteract the division of the original task" (pp. 18–19). This is shown through the inflection point. When around six or seven people start to become involved in the incident response process, the amount of time required per incident increases sharply with the addition of further team members.

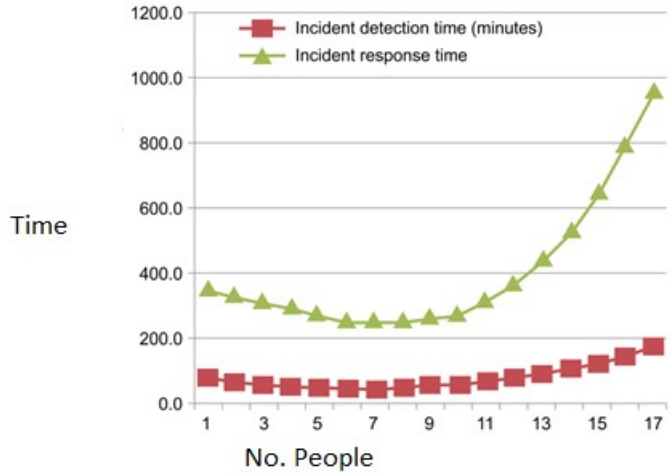

Figure 34. Incident detection vs. response times.

This holds in both response and detection time. Additional people help an incident response team to a point. Adding the system administrator, a coordinator and other such parties does reduce the time per incident, but only to a point. At this point, the effort to coordinate team members starts to negatively influence the expected gain.

Figure 34 shows that additional team members beyond the optimal number have a greater negative effect on the incident response time than on the detection time. These differences can be used to create the teams that are customised towards the needs of an individual organisation.

5.3.1 The Economics

The primary issue at stake is organisational economic impact, though it is more difficult to quantify overall.

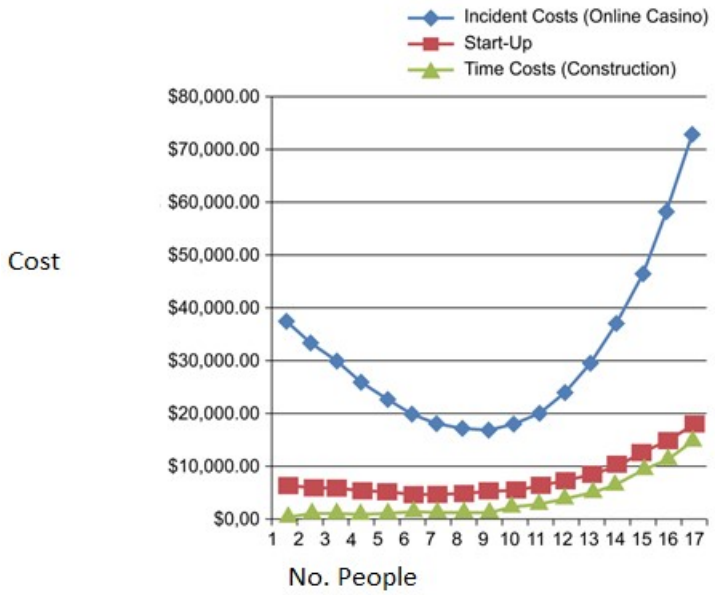

Figure 35. Costs by organisational class.

Figure 35 compares the mean costs of an incident (in consulting fees, displaced revenue, etc.) for several types of organisations. In this case, the online casino operation shows the greatest impact of under-staffing its incident team. Conversely, a construction firm with little existing online presence may acquire little benefit from doing anything (Figure 36). In this instance, given a lack of regulations (e.g. PCI) and other restrictions and an open form style design for blueprints for construction firms, there was little incentive for the firm to care about security. In this instance, doing anything would incur an economic cost. This, however, is the exception as only two of the 165 organisations displayed this pattern.

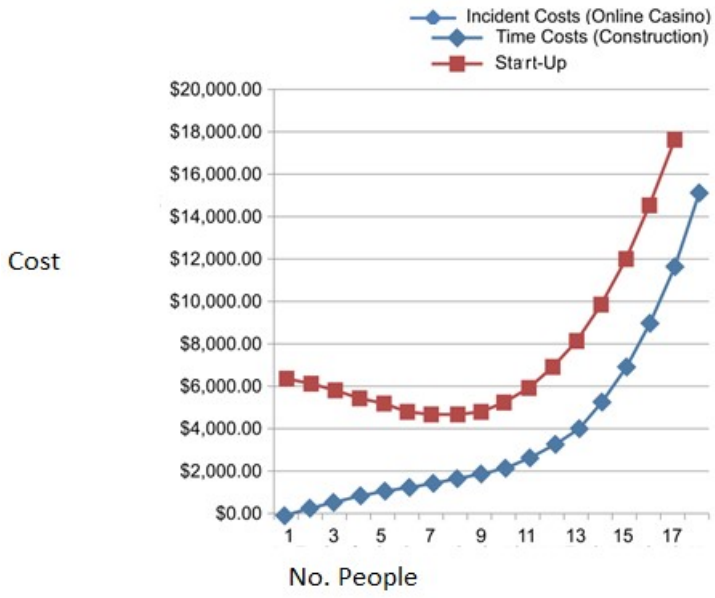

Figure 36. Costs against responders for selected examples

One can thus deduce that it is best to determine the costs and impact of incidents for an organisation and construct a team suited to the needs and requirements that it faces before an incident occurs.

5.4. Using Checklists to Make Better Best

It is reasonable to conclude that with improvement in proficiency at a given task, checklists and reminders for help become unnecessary. This chapter presents evidence that demonstrates the presence of a prejudice against using checklists in the belief that it is harmful to incident response. It also shows that the creation of a simple checklist of steps before an event occurs will enable an incident responder to minimise the number of errors and outlier events.

Gawande (2009) showed that a general reluctance to utilise checklists prevails across multiple professions. In Gawande's work, several

examples and studies was cited demonstrating how the use of a straightforward checklist could improve the results of common medical procedures and save lives.

Considering the wide range of security papers calling for the use of checklists (Baskerville, 1993; Martin, 1973; Dhillon & Backhouse, 2001; Denzin & Lincoln, 1998), remarkably little research is available quantifying the effects of using checklists to reduce the risk to which a system is exposed. Checklists have evolved over time and proponents of lists (Elberzhager et al., 2009) have created checklists for about every conceivable situation.

Bishop and Frincke (2005) stated that, "security checklists cause both alarm (because they can replace thinking with conformance to irrelevant guidelines) and delight (because they serve as aids in checking that security considerations were properly taken into account)", and demonstrated that checklists can boost security testing. Bellovin (2008, 2009) asserted that a poorly structured checklist, "especially if followed slavishly or enforced without thought—can make matters worse."

In the study, the individuals could create their own checklist, both to minimise any potential aversion to using such a tool as well as to reflect the best practices of the individual and organisation.

5.4.1 Methodology in the Checklist Experiment

The experiment was created as a simple analysis to minimise any impact on existing operations in firms that allowed the involvement of their personnel in the study. The names of the firms have been withheld in accord with the firms' wishes and in the general scholastic interest of maintaining anonymity of study participants (organisational or otherwise.)

The experiment conducted an analysis of incident response personnel as they reacted to incidents, comparing response times with and without a checklist in the same individuals. All the individuals were highly skilled and maintain industry certifications such as GCIA and GCIH related to their roles. The manager in the leading organisation was 6-Sigma trained and was receptive[lxx] to the experiment. When the test was started, none of the individuals used a checklist.

In the study, everyone had to create a checklist of at least one and at most two pages in length. A flow diagram was allowed. The analysts were asked to create their own checklist using their own processes, based on the steps the responder would expect in any normal incident. The experiment required that the analyst had to use, step through and read the checklist at the start of the incident when the checklist was used. If any significant event occurred during the incident process, the checklist was to be read again.

The results were measured based on the time taken to respond. A later analysis of the results was conducted based on the recorded data at the organisation, for measuring error rates to improve responder capabilities.

It was expected that the responses of participants would fluctuate, since employees regularly encounter good and bad days at work. In addition, a range of incidents would occur on a given day and over the course of multiple days. To account for this variability's effect on the day-to-day choice to use a checklist, the responder would toss a virtual coin via a small process loaded into the web portal. Based on the result of that coin toss, the system was set to change to show a green icon on the analysts' screen if they were to use a checklist and to present as blue when they were not to use the checklist.

5.4.1.1 To define the variables and hypothesis.

The hypothesis was that there would be no statistically significant difference in mean results between the times taken from the start of an incident or event to the determination that an event had or had not occurred. This is the value measured, defined as time in minutes "*t"*.

If "*t*" was larger for those with a checklist, the result would be negative, suggesting the checklist produced undesirable effects. Conversely, if it was found that "*t*" was smaller for those incidents when a checklist was used, one could infer that a checklist improved the incident response process. In this test, the responders were all highly skilled. It would be expected that any positive results found for a highly trained responder would be more beneficial for an inexperienced security professional or even a more generalised IT professional (Bishop & Frincke, 2005).

The experiment measured the following variables:

- t_{ij} Here i is the i^{th} individual with the value 'j' in minutes to establish a response and determine if an event was an incident or not.
- $t_{ij(check)}$ This is the subset of readings where the individual 'I' used a checklist as measured in minutes.
- $t_{ij(free)}$ This represents the subset of readings where the individual 'I' did not use a checklist.

Using these variables allowed for the calculation of the following:

- $t_{i(ave)}$ This is the average response time in minutes for an individual 'I'.

- $t_{i(check)}$ This is the average time for the individual to respond and determine if an event is an incident using the checklist.
- $t_{i(free)}$ This is the average time for the individual to respond and determine if an event is an incident without using the checklist.

With these values, the study and hypothesis can be defined particularly well. First, define Ho as the null hypothesis and Ha as the alternative hypothesis. The hypothesis is expressed as follows:

- Ho $t_{i(check)} = t_{i(free)}$
- Ha $t_{i(check)} <> t_{i(free)}$

The null hypothesis is that there is no difference in how long it will take, on average, for an individual using a checklist to respond and establish if an event qualifies as an incident or not. The alternative hypothesis is that the use of a checklist *will* result in a difference in how long the responder reacts. This tests whether the time taken to respond using a checklist will be significantly different than without a checklist.

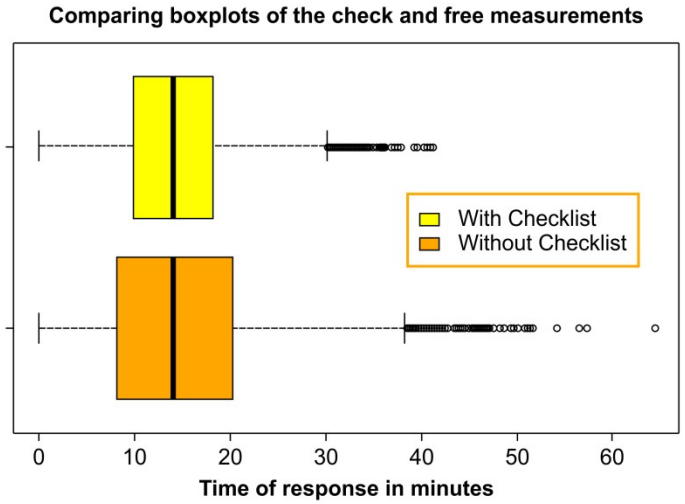

Figure 37. Comparing response times with and without checklists.

Although each event or incident will vary in style and structure and the responder themselves will vary in capacity throughout the day and at

different points in their lives, the averages when taken over time should be the same (United States Fire Administration, 2004). To ensure that the responders followed the process, the responders created and used their own checklists (created within a set guideline) based on best practice as they determined and defined it. These controls removed arguments over best practice and individualised processes that would have been expected to occur.

Responder bias was minimised using a system of randomising the times when a responder would use the checklist. The study did not just set times of the day used in the process. Also, effects from time of day and week were also removed statistically. As a virtual coin toss determined if the responder used the checklist or not, the responder did not select the responses for which they would employ a checklist.

Table 10 A comparison of ticheck (time with a checklist) against tifree (the time without using a checklist.)

	Minimum	1st Quartile	Median	Mean	3rd Quartile	Max
ticheck	0.000	9.878	13.680	14.000	18.000	41.260
tifree	0.000	8.246	14.060	14.360	20.280	64.600

5.4.2 Results

The results of the study are presented in Figure 39 as a boxplot. In visualising the two datasets, it is possible to determine that there is a difference in the standard deviations with a larger range of values for the responses without a checklist than those recorded when a checklist was used. Looking at the statistics in R (the statistical package, displayed in Table 10), one sees a mean (average) value of 14.3602 minutes for responses without the use of a checklist and 14.00188 when a checklist is used.

The mean values differ by only 21 seconds on average over a mean of around 14 minutes. This visual inspection does not show the complete result. A Student's t-test conducted on the two datasets shows whether a difference in the values exists or not. The results are displayed below (as output from R).

```
> t.test(tifree, ticheck)
        Welch Two Sample t-test
data:  tifree and ticheck
t = 3.3964, df = 20638.23, p-value = 0.000684
alternative hypothesis: true difference in
means is not equal to 0
95 percent confidence interval:
 0.1515310  0.5651013
sample estimates:
mean of x mean of y
 14.36020  14.00188
>
```

What this all means is that at the alpha = 5% level, there is a p-value of 0.000684 and we assert that there is a statistically significant difference in the means. Hence, the null hypothesis can be rejected in favour of the alternative hypothesis Ha, that a checklist does exert a positive temporal effect on the diagnosis of an incident.

5.4.2.1 Type I error and monitoring intrusions

The incident analysis and the rate of error in diagnosis were measured using an experiment where noted incidents where replayed. Existing PCAP capture traces from sites with known attack and incident patterns were loaded into an analysis system for evaluation purposes. The OSSIM and BASE frontends to SNORT had been deployed for this exercise. SQL scripts were altered to display a random lag into the responses and TcpDump was used to replay the PCAP trace as if it occurred 'live'. The analyst had to decide if each incident was worth escalating or should be noted and bypassed. The results of this exercise are illustrated in Figure 40 through a display of type I errors.

Figure 38 demonstrates that as the response time of the system increases, so does the analyst's error rate. The lag in returning information to the analyst has a direct causal effect. The longer the lag between requesting the page and that where the page is returned, the greater the error rate in classifying events.

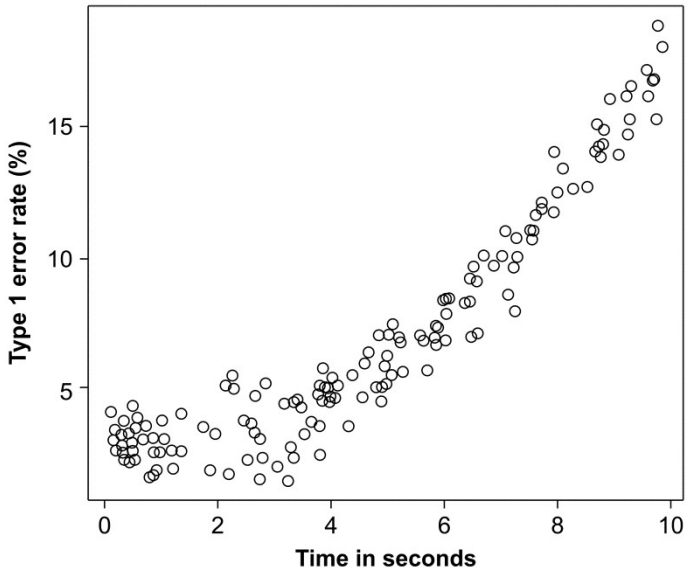

Figure 38. Type I error rate and the time to respond.

The analysis of this data was extended to include a Loess calculated plot of the expected error /rate against time (Figure 39).

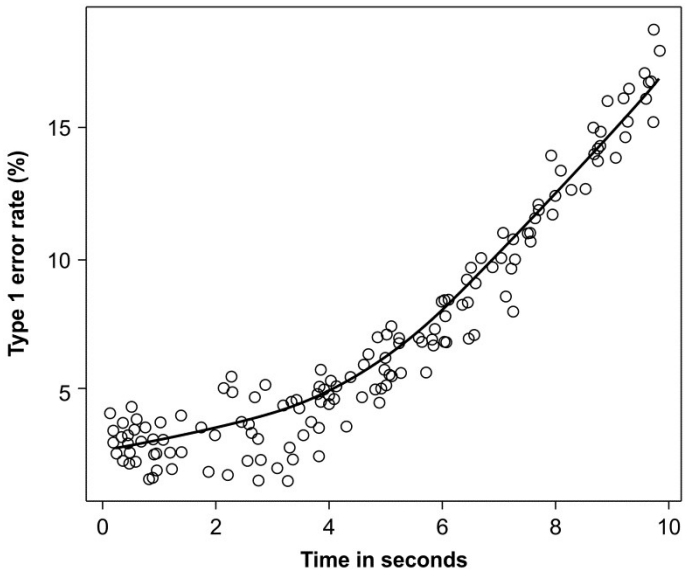

Figure 39. A Loess plot of the Type I error rate for the time to respond.

The plot shows that the slope increases sharply after around four seconds. As such, it is essential that responses to queries are returned in under 3–4 seconds. This suggests that time delays significantly affect the performance of an incident handler.

5.4.3 Discussion

Although there was little difference in the mean value returned from the use of a checklist to that when the responder did not use a checklist, there are some outliers in which something has gone wrong. The boxplot (Figure 37), shows that there are occasions not involving a checklist in which errors do exist. These mistakes increased the time required for the incident response process. There is a long tail effect when no checklist has been used.

In responding to an incident, having a checklist helps even experienced professional incident responders. A professional may have worked in the same capacity for many years and understand his work

implicitly, yet under stress or in a rush, certain things may be missed or overlooked (Hales & Pronovost, 2006).

The null hypothesis can be stated as there is no difference in how long it will take an individual on average to respond and to determine whether an event constitutes an incident. This allowed for the capability to measure and compare the time taken using a checklist against not using one. The alternative hypothesis is that the use of a checklist will result in a difference in how long the responder reacts. The time measured using a checklist will be significantly different from that without a checklist.

When the initial result is coupled with the result that time delays increase Type I errors, using a checklist is an effective low-cost means of improving the security and response times against an attack.

Each IDS system has an expected Type I and Type II error rate that will change as the system is tuned to the organisation's operating environment and with the input of the responder. Consequently, this represents a peculiar function for the organisation that can only be approximated for other organisations (even when the same IDS product is deployed). The inherent accuracy of the IDS (which is a trade-off between Type I and Type II errors and is a cost function) will be dependent on the input of the individual assessing it. The IDS forms a cost function as the increase in reporting results in a greater number of false positives that need to be investigated. In limiting the false positives, the likelihood of missing an event of note also increases. Each validation of a false positive takes time and requires interaction from an analyst. Hence the tuning of an IDS is balanced on maximising the detection rate against cost. Adding reasonable controls that reduce error has value and improves the security of a site.

It is easy to see that the more routine a job is, the greater the need for a checklist (Turner, 2001). Even the smartest of us can forget where we parked our cars on returning from a long flight (Mann 2002). So we should be asking why not make a basic checklist that will make things better? In Information Technology operations, the vast majority of knowledgeable people have re-built servers, but in an incident response environment, it can be unforgivable to overlook a serious security configuration simply because the stress of the environment can cause one to lose track of which stage they were on while being interrupted and multitasking (SANS 2011). We show that the use of straightforward checklists and flowcharts created by the individual make for better results even in daily tasks.

5.5. Rewarding IT Staff in a Changing Environment to Promote Secure Practice

Lane (2004) stated that "once an organisation has selected its employees … it will attempt to find some means to measure and appraise their performance" and that "in the absence or presence of a formal appraisal system, informal appraisal of work and behaviour takes place continually". Security is as much a function of personnel as technology. As such, the management of risk is a process of managing and incentivising individual employees. The following section will first look at performance management and the effect of behaviour on performance from a theoretical viewpoint.

Next, the section addresses reward management as it pertains to information technology staff and in incentivising risk effective behaviours. This is explored by contrasting performance management

techniques and fairness in the organisation, the team and the individual. Management can encourage productivity by rewarding those behaviours that reduce the risk of a security compromise. Alternative methods to reward are investigated as more effective and productive alternatives to the typical performance appraisal scheme.

5.5.1 Defining Performance Management

Williams' (2002) performance management system starts with input such as a high end managerial statement (such as corporate policy or performance plans), a control mechanism for formulating and overseeing performance objectives, and finally a series of controls used to evaluate and remunerate outputs with respect to products and/or services in a fair and effective manner.

Lane (2004) postulated that the precise nature of performance management remains "indistinct". He argued that this can explain why "many textbook writers use the term performance appraisal and performance management interchangeably, as if these concepts were one and the same thing". Competing definitions of performance management have led to a discussion of performance appraisal (Armstrong, 1994) rather than performance management, and a conflation of the two terms (Lane, 2004).

Given the need to measure and report on the security of data in modern enterprises, the ability to create effective metrics and measure employee awareness and compliance with security policies and practice within an organisation needs to become a key aspect of performance management. Yet, if management researchers cannot agree on the nature of the discipline, the creation of a set path towards measuring employee awareness becomes less viable.

Williams (2002) argued that, “even at its most basic, performance management isn’t about a single intervention”. Yet Lane (2004) stated that “one main interpretation of performance management continues to dominate practice” with the management of people based on individual interventions. Nankervis and Leece (1997) concluded that performance management systems meet the needs of management but not the needs and welfare of those most directly affected by performance management. When the primary focus of performance management is the individual, management often fails to address a continuous process of improvement and organisational performance. The result is to allow blame to pass to the individual and not the structure that has permitted a control failure or breach.

5.5.1.1 Performance management starts when the employee commences

Orientation is a common method of introducing a new employee to the organisation. For orientation programs to be effective, new employees must receive specific information about the following three areas (Lane, 2004):

1. Company standards, expectations, norms, traditions, policies;
2. Social behaviour, work climate, getting to know colleagues and supervisors;
3. Technical aspects of the job.

Lane (2004) noted that orientation occurs at two levels, “the company (conducted by HR representative), and departmental (conducted by direct supervisor)”. It was further noted that a successful orientation program includes a process of follow-up and evaluation. Many organisations also use this process to impart their security policy and practice.

Competence development in employees is a primary goal of performance management (Williams, 2002), requiring an effective training program that includes assessment, implementation and evaluation. Performance appraisal interviews, a fundamental component of performance management for many organisations, rely on methodical portrayals of the job-relevant strengths and weaknesses of individuals in the group in order to improve the professional performance of the employees and to disseminate information to management for use in future decision making (Thompson & McHugh, 1995). It is further argued that any dysfunctional aspects of managing employee performance may be solved by a study of organisational behaviour. Continuous feedback could be used as one alternative method of addressing personality conflicts and employee performance monitoring. In terms of risk management, this strategy must make employees aware of the security concerns faced by the organisation.

Thompson and McHugh (1995) further posited that performance management should reduce the dysfunctional aspects of the interactions between intraorganisational entities while simultaneously "reinforc(ing) the positions, rewards and activities of dominant groups in organisations (managers)". This requires that the leading groups in the organisation actively embrace the security governance efforts if the organisation is to embrace these as well. That is, a top down approach.

Walker (1992) details an interlinking framework of strategic contexts/expectations, performance objectives, work, coaching (or mentoring), and training designed as a control process. Personal performance, abilities and knowledge mixed with equitable rewards, and motivation (or reinforcement) coupled with performance feedback will bolster performance and motivation. Incentivising secure behaviour that

still allows employees to fulfil their assigned duties is the most efficient means of creating a security-aware culture.

This view formulates a strategy to reward positive behaviour and discipline negative behaviour to modify employee output. These rewards (such as promotions, increases in pay, and training opportunities) and punishments (demotion, negative feedback or dismissal) are used by the organisation's management to shape its workforce and gain an overall acceptance of risk governance. This progression is intended to strategically advance the organisation by improving its competitiveness.

5.5.1.2 Individual Performance

Individual performance is characteristically the focal point of a performance management system and is also the focus of many IT and, in particular, information security staff (Myers & McCaulley, 1985). Traditionally this involves staff being held responsible for the key result areas and outputs of their job description (Lane, 2004) where "job descriptions are criticized as being inflexible, static, rigid definitions of responsibilities, and are probably inappropriate for turbulent work environments". Job descriptions of this style often ignore multi-skilling for teamwork and have led to an individualistic haven for information security staff seeking an independent view of the organisation.

IT staff often seek a clear notion of individual accountability (Myers & McCaulley, 1985) that conclusively establishes the work to be done, results to be attained and the attributes (skills, knowledge and expertise) required to achieve these results; or, in other words, "how will I know I have achieved what I set out to do?"

5.5.1.3 Team Performance

Sheard and Carbone (2004) noted the evolution of IT environments towards a grouping of staff members with less independent and individualistic needs.[lxxi] Unfortunately, individual performance appraisal systems are often poorly suited to such team-based structure (Lane, 2004), focusing on individual endeavours as opposed to measuring and rewarding team performance.

5.5.1.4 The organisation as a whole

Locke and Latham (1990) have been noteworthy in demonstrating the impact of goals on work performance, revealing that difficult challenging goals lead to higher performance than easy goals, if the job holder accepts and is committed to the goals. Additionally, it is noted that specific goals lead to higher performance than do vague, general "do your best" goals, or no goals at all. However, it is imperative to set specific targets, in security as in any workplace function.

Consequently, when applied to the organisation, security goals need to be team orientated, challenging and specific (Ghoshal, & Bartlett, 1995). Most importantly they also need to be aligned with the requirements of both the business stakeholders and the employee. This difficult task is prone to falling back on an individual performance appraisal system without delivering the benefits it promises.

5.5.2 Reviewing and Supporting Performance

According to O'Neill & Kramar (1995), "pay and benefit costs are the single largest operating expense … Pay for performance … promote[s] a unitarist rather than a pluralist approach to employment" in rewarding the efforts of the individual over those of a collective in pursuit of bolstering security. Stone (2002) defines merit pay as "any salary increase awarded

to an employee based on their individual performance", a definition supported by Williams (2002a). As the nature of information security work is traditionally creative and individualistic, the performance structure must be devised accordingly. Moreover, incentives need to be tailored in a manner that encourages a group approach to security.

5.5.2.1 Pay by Merit

Merit pay is common in executive and management pay structures (Kramer, et al., 1997) and is intended to "develop a productive, efficient, effective organisation that enhances employee motivation and performance" (Hoevemeyer, 1989). Merit pay is becoming more common in IT as employees are being offered bonuses for successful completion of business projects as well as for meeting operational security targets. When coupled with visible security and risk metrics, bonuses can be used to incentivise secure behaviour, but only when coupled with monitoring systems (Wright, 2011f). When compliance is measured, and rewarded indiscriminately, there can be perverse reductions in the overall levels of organisational security.

Ivancevich and Matteson (1999) asserted that pay by merit schemes do not reward accomplishment when "employees fail to make the connection between pay and performance, [and] other employees perceive the secrecy of the reward as inequity". IT with a largely independent employee base, often suffers as these types of arrangement may be perceived to be unfair, encourage employees to be risk adverse and focused on compliance rather than security per se, and increase distrust between staff and management.

Brown and Walsh (1994) suggested that management is mistaken in assuming that pay is adequately appreciated as recompense and thus acts as a motivator for all employees. Two-factor theory (Herzberg et al.,

1959) classifies pay as a hygiene factor: it alone does not motivate the employee, but its absence can prevent other sources of motivation, such as recognition, responsibility and advancement, from occurring. Hence, other forms of incentives may be more effective at promoting a corporate culture of security.

Strategic human resources management is required to accurately determine an effective reward program, one that involves the "measurement of productivity, performance appraisal, training, performance-related pay (Lee, 1996), profit-sharing and share ownership schemes, and job redesign, with a management philosophy that espouses teamwork, consultation, communications and information sharing" (Bamber et al., 1992).

Alternatives to the conventional model of employee financial compensation include:

- Performance-based pay,
- Competency-based pay,
- Broadbanding (moving a large number of employment grades with narrow salary bands into a structure with few broad grades using wide salary bands (Stone, 2002),
- Team-based pay,
- Employee share/option or recognition schemes (O'Neill & Kramar, 2003, p. 196),
- Value-added packaging (including laptops and training plans that form an extended remuneration package).

In encouraging staff to promote security, alternatives to pay should be considered. Value-added packaging is commonly used in rewarding IT staff and this can be extended within organisations.

5.5.2.2 Training

"Training personnel to acquire knowledge, skills, and attitudes is an essential role for instructional systems design, and so is training that translates knowledge, skills and attitudes into effective performance" (Davies, 1994, p. 111). Jackson (1992) asserts that employee training should not just be seen as a means of improving performance but as a means to promote a security-aware culture. Training can be both reward and ambition to the IT employee such that the development that training brings is itself a reward.

5.6. Human Resources Management: Hiring the Right People to Remain Secure

Human resources represent, in brief, a short-term cost that earns long-term gains. Ultimately, effective management of human resources (in business in general and in the IT field in particular) represents one of the most effective risk mitigation measures (Kovacich, 2003). Christopher (2003) asserted that a lack of training can lead to employees making "one of the worst mistakes" and "giving out sensitive data". Training and education empower staff to make correct decisions.

From the perspective of the human resources professional, the key sections which need to be addressed to create a secure environment include:

1 Segregation of duties,
2 Recruitment, and
3 The monitoring of personnel.

Many of these controls are essential requirements for either COSO or COBIT and thus normal compliance and audit efforts, so human resource management (HRM) is an essential part of achieving IT governance. Some researchers (Banerjee et al., 1996) asserted that HR management is not only a stage in IT governance, but is essential to preventing breaches of corporate security being caused as a result of "weakness in human firewalls". This underscores the need for awareness training for staff, as technology will fail when staff are not fully educated in stopping attacks against the organisation's information infrastructure.

Training and technology need to work in conjunction to protect security. This requires horizontal integration and cooperation among IT, HR and department heads. One key issue at stake is that management needs to educate and communicate the corporate policy, across the entire organisation (O'Brien, 1999). This requires that management implement feedback channels within the organisation (Mitnick & Simon, 2002). Turnbull, I (2004) argued that organisations face new challenges and that they need to plan for these to be successful. Best practice is only achieved through a process of knowledge and empowerment across all staff. In this, new domestic and global HR privacy demands are driving many of these changes, adding yet another layer to the security framework and the compliance regime.

5.6.1 Defining the roles, HR needs to work with Info Sec

Kovacich (2003) presented a total systems approach to all the elements needed for the "infosec professional", of which human factors are among the most critical. He asserted that defining the position of the information systems security officer (ISSO) is just a beginning. Poor hiring and HRM practices adversely impact the security of the organisation and lead to

unnecessary risk. Compliance is just the foundation for HR security controls (Dhillon, 2001).

One concern related to HRM practice has resulted from a widespread shortage of security, audit and compliance skills (McCarthey, 2001). The security compliance drive has stirred debate amongst many professionals (not just those from HR) over the practice of hiring criminal hackers, who thereby become the proverbial "fox in the henhouse" (Savage, 2003).

On the question of hiring hackers, "trust should be evaluated on a case by case basis", advised Mitnick (as cited in Savage, 2003), the president of an information security firm and past convicted "hacker." This risky yet often effective practice only highlights the need for well thought out, systematic security practices within an organisation.

Good hiring policy, detailed background checks and controls should all be designed to increase the chances of hiring the correct person for the role and ensuring that they remain satisfied (Fitz-enz, 1997) and perform well (Wood, 1995, 1997). This creates a series of processes that help reduce risk and improve efficiency within an organisation. If an organisation is to minimise the risks faced through human security leaks, it is important that HRM and information security teams work closely.

5.6.2 Awareness

Dhillon (2001) stated:

> "Education, training and awareness, although important, are not sufficient conditions for managing information security. A focus on developing a security culture goes a long way in developing and sustaining a secure environment…a mismatch between the needs and goals of the organisation could potentially be detrimental to the health of an organisation and to the information systems in place… organizational processes such as

> communications, decision making, change and power are culturally ingrained and failure to comprehend these could lead to problems in the security of information systems."

Mitnick and Simon (2002) stated that there are three key steps that should be instilled within every employee's thought processes:

- Step One: Verification of Identity,
- Step Two: Verification of Employment Status,
- Step Three: Verification of Need to Know.

They further stated that deceptive tactics are generally used to access or obtain private company information by masquerading as a trusted party. For this reason, it is essential to verify the legitimacy of employees, contractors, vendors, or business partners. Effective information security is maintained only if an employee receiving a request to perform an action or provide sensitive information must positively identify the caller and verify his authority prior to granting a request.

For this reason, a well-rounded awareness program must cover as many of the following key areas as possible (Wilson & Hash, 2003):

- Security policies related to systems passwords (these include computer and voice mail),
- The procedure for disclosing sensitive information or materials,
- Email usage policy, including the safeguards to prevent malicious code attacks including viruses, worms, and Trojan Horses,
- Physical security requirements such as wearing a badge,
- The responsibility to challenge people on the premises who aren't wearing a badge,
- Voicemail usage in accord with best security practices,

- A systematic means of determining the classification of information, and the proper safeguards for protecting sensitive information,
- Proper disposal of sensitive documents and computer media that hold or have at any point held confidential materials.

Additionally, the awareness program relies on the following tasks to be successful:

- The development and distribution of an IT security policy that reflects business needs tempered by known risks,
- Informing users of their IT security responsibilities, as documented in the organisation's security policy and procedures, and
- Establishing processes for monitoring and reviewing the program (Ciocchetti, 2010).

Wilson and Hash (2003) stated that firms must explain the appropriate conventions of conduct for the use of the organisation's IT systems and information. HRM is crucial as changing peoples' attitudes and behaviour in terms of IT security can be a challenging task. New, prudent controls sometimes conflict with the way staff have done their job for years. An awareness and training program executed by HR can enable employees to adapt to such changes, remaining flexible and dynamic enough to guard against ever-evolving threats.

Coe (2003) argued that "recurring evaluation and maintenance of employee awareness, specialized training and management awareness are all required components of a successful security program". To be effective, an information security program needs to properly account for the strengths and limitations of employees to successfully secure an organisation's data. So, in order to maintain security and keep a "network

safe, HR must protect sensitive data from internal and external security threats" (Romeo, 2002).

5.7. Chapter Conclusion

Using a checklist is beneficial in an incident response. The experiment presented in this chapter demonstrated that the use of straightforward checklists and flowcharts created by the individual employee yields better results even in daily tasks. That stated, it is also crucial to ensure that checklists are flexible and do not lead to rigidity and a lack of innovation. The development of a checklist can be made a task of the individual doing the task and in any event, must be updated to reflect a changing environment.

Human resources management is an often overlooked, but essential component of information security within an organisation. Information security personal and human resources need to work together to ensure the overall effectiveness of controls. Technology alone is not a panacea; instead, it is most effective in a dynamic, symbiotic, and security-aware relationship with personnel and the employees who manage them.

One can thus deduce that it is best to determine the costs and impact of incidents for an organisation and construct a team suited to the needs and requirements that it faces before an incident occurs.

The creation of a security department is a function of both individual and team performance and the careful balance of the right people and training is demonstrated to make a significant impact. However, as with all economic functions, the returns diminish and the aim of a Human resources function in creating a secure environment is balancing the

correct mix of people, training and rewards to ensure that the returns and investment are correctly aligned.

Hence, to create a secure system, there is a need to ensure that the people involved with the configuration, use and maintenance of the system are incentivised to maximise the value of the security investment.

Chapter 6 Modelling Crimeware

6.1. Introduction

This chapter starts with the proposition that organised criminal groups can be modelled using rational choice theory (Wright, 2011b). It is proposed that criminal groups act as profit-seeking enterprises, and the ability to shift economic returns away from this activity results in a lower amount of crime. Criminal behaviour does not need to rely on social deviance, but is moving towards the greatest financial rewards. Consequently, as criminal groups face few financial disincentives, a growing class of criminal specialists has resulted. The course of action to best minimise the online criminal threat can be linked to minimisation of economic returns from cyber-crime (Lynam & Potter, 1997).

To support this, a model is presented that extends the concept of economic defendability (Brown, 1964) to the use of criminal botnets. The research shows that the size of a botnet is constrained in economic terms. Crime groups need to maintain "territories" based on compromised systems (Wright, 2012b) that need to be defended from competing criminal groups as well as from the system owners who seek to purge the attacker as an immune system seeks to purge a virus.

In looking at the types of systems, one can compare the time required to maintain the botnet against the benefits received and then formulate economic defence strategies that reduce the benefits reaped by the botnet. I look at the decision to be territorial or not from the perspective of the

criminal bot-herder. This is extended to an analysis of territorial size. The criminal running a botnet seeks to maximise profit (or economic returns). In doing this they need to analyse the costs expended and benefits received against the territorial size. The result is a means to calculate the optimal size of the botnet and the expected returns. This information can be used to formulate security strategies designed to reduce the profitability of criminal botnets.

6.2. Criminal Specialisation as a Corollary of Rational Choice

6.2.1 Cyber-crime as a rational choice

As demonstrated earlier in the thesis, criminal behaviour can be explained, in part, by the rational choices of *Homo economicus* in pursuit of economic goals. Rational choice models can be created in order to study some of the decision processes used by cyber-criminals (Pillai & Kumar, 2007). This can pertain to the levels and type of activity as well as the target of an attack. The choice of actions (DDoS, intrusion, etc.) can be coupled with the selection of individual or multiple targets (Bednaret al., 2008). Starting with an initially optimised model, various aspects of information security and cyber-crime can be investigated through the modification of the model parameters such as earnings, costs and preferences.

Whether taken as an individual or criminal group, decisions should be made concerning the allocation of revenue (R) across criminal actions (C) and the composite good (G). This dynamic is represented in Figure 42. In these examples, G is measured as a (real) inflation-adjusted expenditure for the goods that could be substituted for the expenditure on criminal

activity (whether through direct expenditure or when lost earnings are substituted) against other desirable goods (shelter, food, etc.).

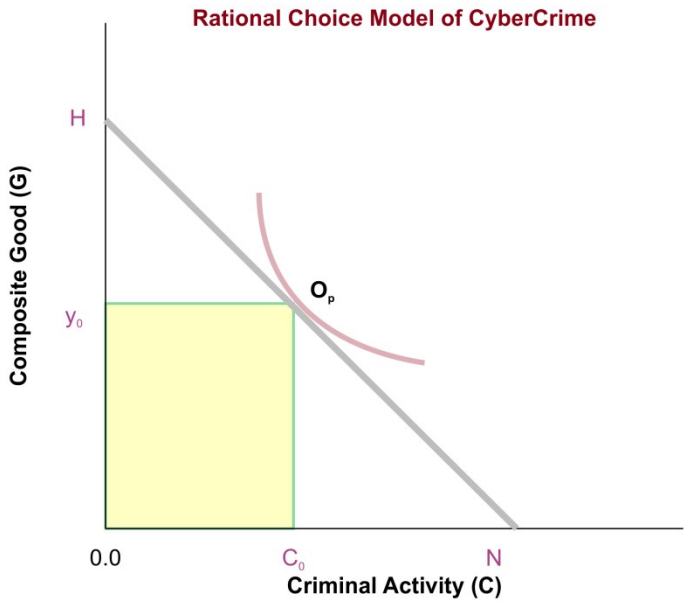

Figure 40. Optional choice modelling for a cyber-criminal of activity against composite good.

In all instances, the cyber-criminal is constrained by economic limitations. Time is a necessary condition in the attainment of goods. This results in the creation of constraints on the most unconstrained attackers. Even the socially inept high school hacker is limited by time. The attacker's economic constraints can be modelled by the limits, $H \leftrightarrow N$ as displayed in Figure 40. This constraint can be expressed as,

$$P_G.g + P_N.n = R \qquad \text{Equation (74)}$$

In (1), P_G is defined as the alternative good that the attacker could have selected or the cost of investing in future attacks. P_N is the averaged per unit cost of completing an attack. The attacker's preference in selecting between G and N is represented in the standard manner by the excepted indifference functions. A negative slope to the indifference curve

represents a propensity of the attackers to sacrifice current goods for an investment in a payoff from ongoing attacks. The larger the slope of the indifference curve, the greater the inclination of the attacker to engage in criminal activity. As the curvature of the indifference function increases, the degree of substitutability between G and N decreases. For a rational player, the optimum decision point O_p occurs at point (c_0, y_0) where the marginal rate of substitution between N and G equals the relative cost of the attack, P_N / P_G .

An attacker also has a choice of targets, T_1, T_2, of which the total resources R must necessarily be divided. In this case, the economic cost equation becomes,

$$P_1.T_1 + P_2.T_2 + ... = R \qquad \text{Equation (75)}$$

hence

$$R = \sum_{i=1}^{n} P_i.T_i.$$

where the number of targets is defined as the value "n" and the cost of attacking each respective target is defined by P_i . The indifference curves and the associated values for multiple targets are individually suboptimal. In total, the sum of all individual activities N_i against the respective target T_1 sums to the total choice of criminal activity at point N no matter how many targets are selected.

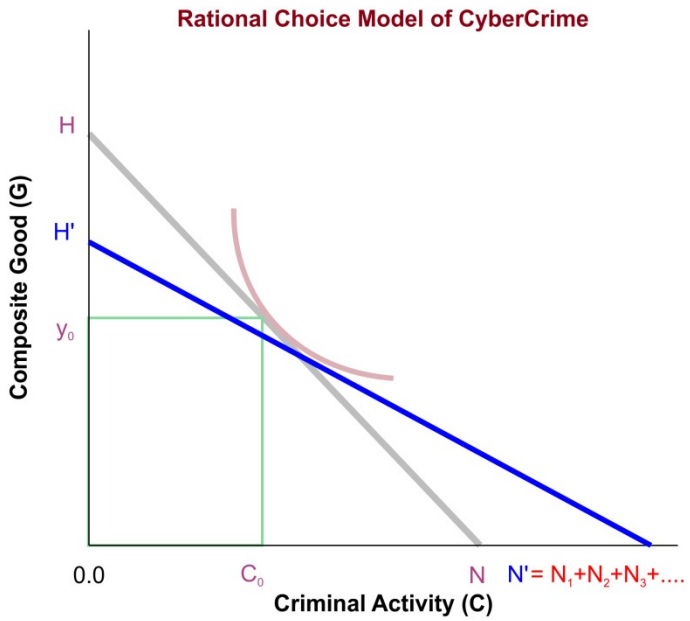

Figure 41. Optional choice modelling for a cyber-criminal of activity against composite good as preferences change.

The attacker can either choose to divide the amount of individual effort and resources across multiple targets by limiting the effort on any individual target or reduce the amount of composite goods that the attacker consumes. By choosing to sacrifice, the attacker alters the amount of criminal activity that they can engage in from N to N' (see Figure 40).

The addition of income from either an external source or the successful completion of prior attacks moves the attacker's budget constraints from HN to H'N' (Figure 41), allowing the attacker to consume a greater quantity of composite goods as well as increasing the amount of criminal activity in which he engages. For instance, the additional financial resources allow the attacker to recruit resources and to develop new exploits and tools (including rootkits, malware, etc.).

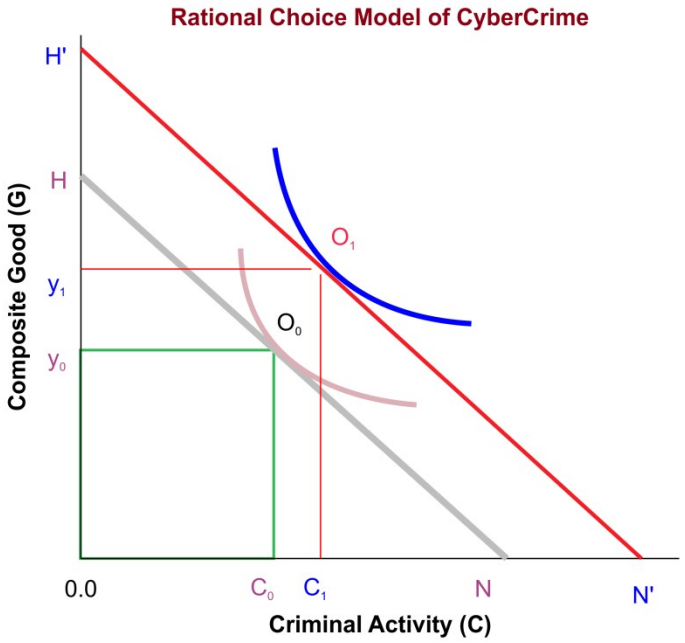

Figure 42. The addition of income achieved by successful cyber-criminals changes the levels of economic constraints and leads to increasing levels of crime.

Conversely, processes that limit the resources available to the cyber-criminal move the criminal's budget constraints from H'N' to HN and thus reduce the amount of criminal activity that can be conducted. These processes include actions that freeze the assets of criminal groups, disrupting the flows of funds (Broadhurst, 2005) and the capacity to trace settlements such as through the restraint of money laundering (Richards, 1999). Furthermore, the limitation of cyber-criminal activity limits the returns on investment, further limiting future investments in criminal activity (Morash, 1984).

6.2.2 Price Change and Cyber-crime

Organisations defend themselves through a process that may be thought of as price policies. The addition of defensive technologies (such as firewalls) increases the price or unit cost of conducting cyber-crime P_N. The limitations restricting the ability of organisations to respond to an

attack through some form of retaliation constrain the ability to increase the price controls against cyber-crime. Deterrence in an organisation has a dual effect: first, it increases the opportunity cost of attacking the organisation, and second, it decreases the probability of successfully attacking the organisation, θ. Where few organisations take adequate deterrence measures, the effect is that an attacker becomes less likely to attack the organisation and moves to an easier target or limits the total number and effort per target. In cases where deterrence is common, the opportunity costs for all organisations (Gambetta, 1988) related to an attack are increased.

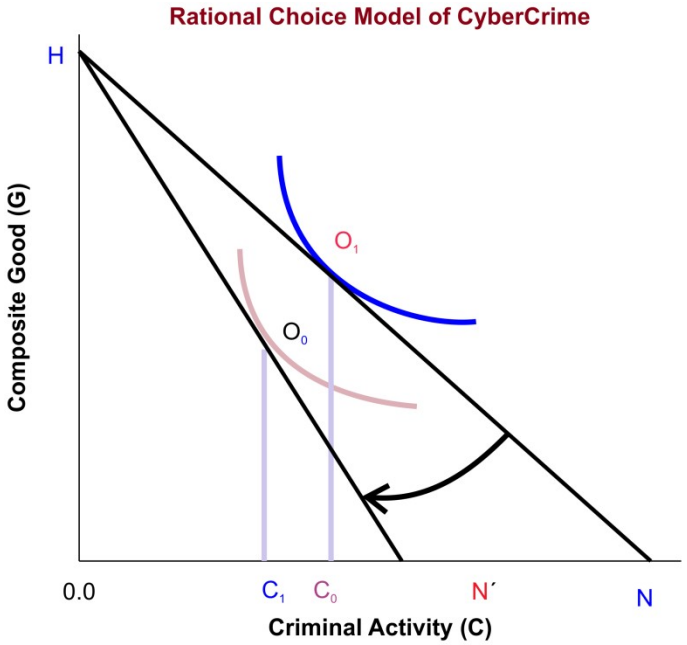

Figure 43. Economic constraints and price policies through increases in the expected cost of criminal activity leads to lower rates of cyber-crime.

Consequently, the attacker has to incorporate the probability of success into the choice of targets (Badonnel et al., 2007). The alternative to composite goods N expended in an attack is expected to return a larger final return of composite good H. Thus, the expected amount of criminal activity C_0 against an organisation will be performed if the expected

return from the activity equals $\left(1/\theta\right)$ times the expense. This may be represented by,

$$\begin{cases} \left(1/\theta\right)(1+\beta)C_0 > E(y_0) & \textit{Attack} \\ \left(1/\theta\right)(1+\beta)C_0 < E(y_0) & \textit{Consider} \end{cases} \qquad \text{Equation (76)}$$

Consequently, an attacker may be constrained by limiting the expected return from the attack $E(y_0)$ to be less than $\frac{(C_0+\beta C_0)}{\theta}$. Where this is achieved, the cost of the attack exceeds the expected returns, and renders an attack unprofitable.

The introduction of additional deterrence costs through the addition of security controls shifts the criminal activity from HN to HN' (Figure 43). The steeper slope of the budget line reflects the additional opportunity costs of an attack. This can be expressed as the decreased probability of an attack succeeding where the previous chance of success θ becomes $(\theta-\varphi)$, such that the new controls reduce the likelihood of an attack succeeding by φ. A consequent expenditure in additional security controls moves the aggregate amount of criminal activity experienced from C_0 to C_1. This is expressed by equation 77 and is consistent with the economic law of demand.

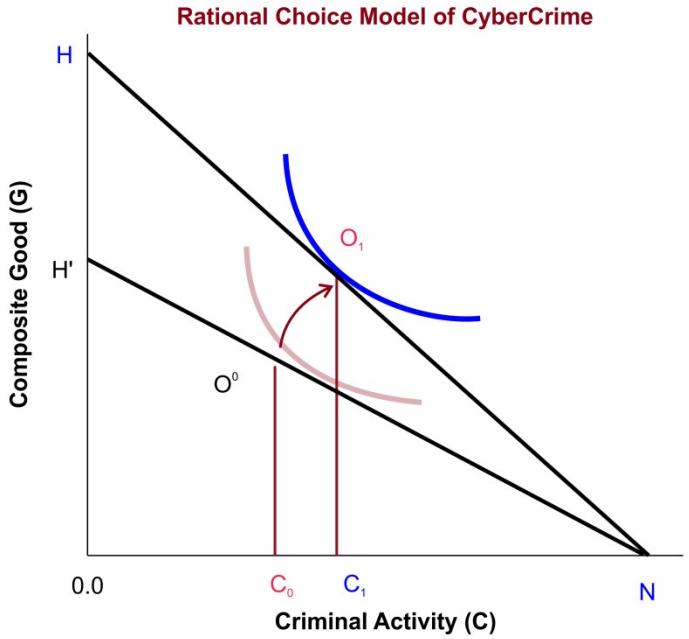

Figure 44. Economic constraints and price policies can create a steeper budget line and reduce the overall level of cyber-crime.

The introduction of additional deterrence costs through the addition of security controls shifts the criminal activity from HN to HN' (Figure 43). The steeper slope of the budget line reflects the additional opportunity costs of an attack. This can be expressed as the decreased probability of an attack succeeding where the previous chance of success θ becomes $(\theta-\varphi)$, such that the new controls reduce the likelihood of an attack succeeding by φ. A consequent expenditure in additional security controls moves the aggregate amount of criminal activity experienced from C_0 to C_1. This is expressed by equation 77 and is consistent with the economic law of demand.

$$\frac{C_1(1+\beta)}{\theta-\varphi} = \frac{C_0(1+\beta)}{\theta} \qquad \text{Equation (77)}$$

hence,

$$C_1 = \frac{(\theta-\varphi)C_0}{\theta}$$

Likewise, any practice that increases the opportunity cost of cyber-crime will result in a reduction of criminal activity. This occurs through an increase in the cost of accessing the common good. Anti-money laundering policies, for example, lower the availability of ready cash (Broadhurst, 2005). This causes the costs of the common good to rise in respect of the proceeds of crime and thus creating an inflationary effect against the criminal proceeds. The overall consequence is the creation of a steeper budget line with reduced rates of cyber-crime.

6.2.3 Game Theoretic Models of Appropriation and Exchange

If one assumes that player A's resources are secure but player B's resources are vulnerable to attack, player A is cast as the attacker/predator and player B the defender or prey (Fowler & Nesbit, 1995). Government bodies may be able to act against player A (through legislation, proceed of crime constraints, anti-money laundering, etc.), but most companies and other organisations cannot engage player A other than through defensive actions. If player B does take vigilante action, they then also become liable to government responses against criminal activity.

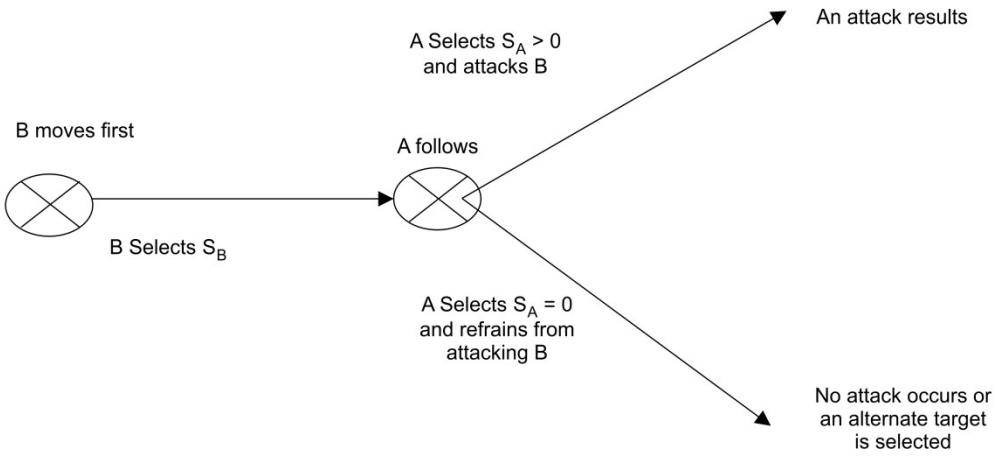

Figure 45. Predator–prey games as a model for cyber-crime.

The possible outcomes of the relationship between A and B include fighting over resources (player A attacks player B) or A and B avoid fighting (A could attack an alternative target or engage in alternative activities). A peaceful settlement of an attack is also not possible as negotiation with criminal groups is illegal and ineffective (such as protection money in mafia-controlled communities (Gambetta, 1991; Gambetta, 1993). Settlement is ineffective as cyber-crime groups cannot effectively stop actions from alternative criminal groups (as might occur in a controlled physical community (Gambetta, 2000).

Player B (the company) makes the first move by diverting resources into security controls S_B that could be used to invest. These controls are thus used to defend player B's remaining investments. A schematic of the predator-prey game is depicted in Figure 45.

Player A moves after B and gauges player B's position (reconnaissance or social engineering can be used to estimate the security controls at B). Player A either attacks B, diverting some of its resources into attacking the security controls of B ($S_A > 0$), or relents (which could involve attacking another target or engaging in legitimate activities).

The decisions of A and B when taken together result in either an attack against B or A engaging in alterative activities and are displayed in the upper and lower branches of the game tree (Figure 45). As B makes the first move, player B can anticipate player A's reaction and choose a level of expenditure on security controls that selects the outcome. When it is profitable, player B chooses a level of security controls that induce A to either attack an alternative (a party who has spent less on controls than B) or to engage in legitimate activity (such as trade). If an attack occurs, the expected loss is set by the multiplier δ .

Where it is not possible for player B to implement a level of controls that ensure A does not attack, B will select a level of security controls that minimises its loss from an attack by A. If an attack ensues, the security success functions;

$$p_A = \frac{S_A}{(S_A + ZS_B)}, \quad and \qquad \text{Equation (78)}$$
$$p_B = \frac{ZS_B}{(S_A + ZS_B)}$$

establish the level of player B's loss function, $(R_B - S_B)(1-\delta)$ where the starting level of resources held by player B is defined as R_B and the resulting level of production and hence available common goods has been diminished for all parties. The Z parameter in the equation reflects the existing level of security of B's systems (where $0 \le Z \le 1$).

From (78), one can see that if player B's security controls falls below an equilibrium point determined by the destructiveness of play from A's attack and B's security expenditure, player B's systems will be seen as vulnerable to attack and the only equilibrium result that can occur is an attack by player A. This effect is interrelated to the loss function, as when

player B expends too much on security in the present, future growth can occur such that an attack becomes inevitable (Devost, 1996).

The solution is to increase the cost of engaging in cyber-crime whilst also minimising the expected returns to criminal groups. If all players engage in a level of controls that lower the expected returns by A to less than that of valid trade, the incentives to engage in criminal actions vanish (Einstadter & Stuart, 1995).

6.3. Territorial behaviour and the economics of botnets

Criminals defend territories in cyberspace (Bensoussan et al., 2010). In this virtual environment, it is possible to engage in economically profitable low-risk criminal activities. The boundaries are like invisible lines on the map of the Internet, but they form connected systems analogous to a real-world territorial environment. The component systems that comprise the botnet are defended by aggressive displays and direct attacks, sometimes by defenders, at times by competing criminal groups.

Several different territorial strategies exist for criminal groups running botnets. Each of these strategies has different benefits and costs associated with them and several of them are independent of the others. Some strategies involve the exploitation of high-value targets (including the exfiltration of data) whereas others involve the use of large numbers of systems to amplify low-value transactions (including SPAM transmission and DDOS attacks). Territories are smaller both when they are higher in resource value as well as when they cannot be further secured by the attacker once the machine is owned because they are less defendable and are usually abandoned early when attacked (Figure 46).

6.3.1 Extra-jurisdictional territories

One territorial strategy would be to only attack extra-jurisdictional systems. This would include machines that are located outside of the criminal's legal jurisdiction. The strategy lowers the risk of being caught and hence lowers the cost associated with engaging in this course of action.

6.3.2 Intra-jurisdictional territories

Criminal groups operate within jurisdictional boundaries when the risk of being caught is perceived as low and the rewards are perceived to exceed the risk. This can occur when the benefits of acting within a local jurisdiction exceed the increased risk of punishment that can result from being caught and more easily charged with an offense within a local jurisdiction.

6.3.3 Non-territorial strategies

An alternative approach would be to compromise individual hosts and networks to extract data and leave the system undetected by defenders. In this scenario, the attacker does not create a territory, but moves across systems in a predatory manner somewhat analogous to a roaming carnivore in nature. In this way, the attacker compromises a server, gains access, obtains the resource they seek and leaves after either covering the tracks or being detected. This could be perceived to be a migratory strategy, with the attacker moving from system to system in a constant attempt to exploit vulnerable hosts. Where sufficient vulnerable systems exist to allow the criminal to profitably move from system to system, this strategy will still provide an acceptable return on their investment of time and resources, allowing the non-territorial strategy a means that can work for smaller numbers of skilled attackers.

The non-territorial attacker also acts as a predatory force when systems that have been compromised by territorial criminal groups are targeted. In this instance the non-territorial attacker increases the cost of maintaining access to the territorial criminal, who is required to expend additional time and resources maintaining their territory as well as increasing the likelihood that the system owner will notice the compromise and respond by removing both sets of criminal groups.

Due to the time and resource requirements associated with attacking and successfully exploiting any given system, the costs associated with a non-territorial strategy limit the criminal to attacking high-value resources.

6.3.4 To defend or not defend

Here, “territorial” refers to the context of holding a group of systems for an extended amount of time. An alternative strategy would be to compromise systems for selected engagements. An example of a non-territorial criminal strategy would be to break into a targeted network, exfiltrate information and abandon the system, covering one’s tracks. In this case, the attacker could remove all traces of the criminal activity before moving on to another target. Conversely, where an attacker maintains access to a defensively postured system, access to those compromised hosts entails a cost and they are engaging in territorial activity.

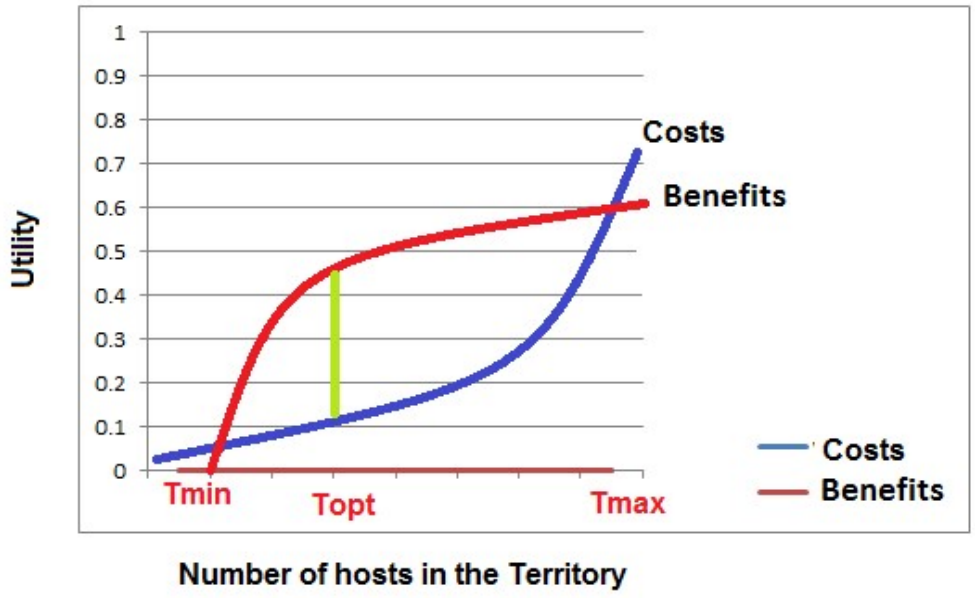

Figure 46. A cost benefit analysis of criminal territory in cyber compromises.

Cyber-criminals must decide whether or not to be territorial, and if so, what size territory to defend (Bensoussan, et al., 2010; Li et al., 2009). Cyber-criminals can engage in either pure or mixed strategies (Clarke & Cornish, 1985). The pure strategies involve either territorial actions, non-territorial actions, or a mixed strategy that incorporates both.

There are multiple costs associated with the acquisition of a territory, whether this is directly related to a botnet or if the system is comprised of otherwise compromised hosts. The attacker needs to not only consider the cost of initially acquiring and compromising the host (Lin et al., 2009) but also the subsequent holding and maintenance of that compromised host.

6.3.4.1 The costs of acquiring resources

The first cost aspect of creating a criminal territory involves the initial acquisition cost. Several stages can be differentiated based on the strategies associated with the individual botnet. These stages include:

- Research,
- Reconnaissance,
- Scanning,
- Exploitation,
- Maintaining access, and
- Covering tracks.

This initially starts with a research and reconnaissance phase. An attacker with a selected strategy will seek out systems to exploit and then seek possible targets.

Each step from reconnaissance to exploitation involves risk and costs to the attacker. Risk increases significantly at the scanning and exploitation stages but there are costs from the initial research stage on. At the minimum, cost can be counted as a time-based resource where it cannot directly be associated and accounted for in monetary terms. The reason for this is that time is a scarce resource with alternative uses. The attacker can make use of the time taken researching and attacking one host for other uses.

Any action taken by the potential target that prolongs the time needed to successfully exploit a system increases the cost to the attacker (Bensoussan, et al., 2010). If frustrated in this way, the attacker could then choose to attack a less well-secured system, change their strategy or engage in non-criminal activities that produce higher returns relative to the time and capital investment required (Parameswaran et al., 2010).

6.3.5 The costs of defending resources

Once a system has been acquired it needs to be defended and exploited by the cyber-criminal (Li et al., 2009). If the cyber-criminal fails to adequately take advantage of the target system (that is, on average makes

a profit per system compromised), they will incur a loss and be discouraged from attacking at a later point (this is shown in Figure 46 at points where the costs exceed the benefits).

Any system that is not adequately defended by the attacker will eventually become a lost resource. This can be modelled as a Poisson decay function whereby the number of hosts held by an attacker diminishes over time. The attacker needs to actively maintain access to the compromised hosts or will in time lose access. Eventually as systems are upgraded or decommissioned by the owners, the attacker will inevitably lose access.

6.3.5.1 Foraging and conflicting demands

The behaviour of cyber-criminals may be influenced by the need to maintain access to compromised systems, scan for new systems, defend territories (both from system owners who will remove the criminal if detected and from predatory criminals seeking to infiltrate and take over existing botnets), defend C&C servers, and so on. While cyber-criminals scan systems, the existing compromised and controlled systems are vulnerable to intruders and predators (e.g., other cyber-criminals or the organisation's security personal attempting to recover the compromised site and system). The result is displayed in Figure 47.

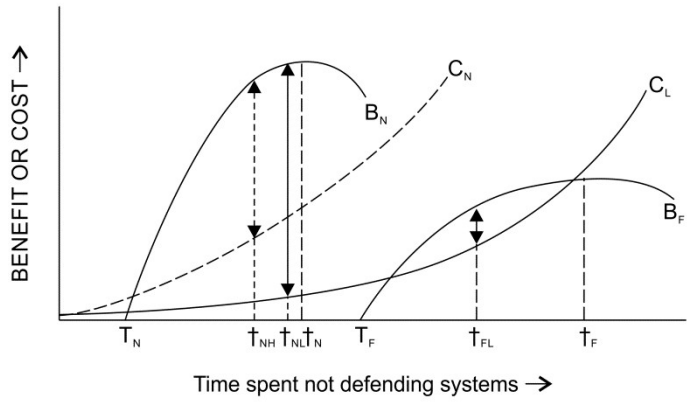

Figure 47. Defence of existing systems limits the recruitment of new compromised hosts.

The terms used in the model (Figure 47) are:

- **N** & **F** = two sets of systems identical except for distance (Pheh, 2008) from the attacker in accessibility (N = near & F = far). A near system would be open to direct access. A far system would require access via alternate paths (such as using a pivot or proxy). Low value systems (e.g. home user systems) will be considered "nearer" than high value corporate systems (which are more likely to be behind a firewall and other security controls)'
- $\mathbf{T_N}$ & $\mathbf{T_F}$ = Scan and attack times'
- $\mathbf{B_N}$ & $\mathbf{B_F}$ = benefit curves (proportional to the rate at which the cyber-criminal recovers costs on attacking a system),
- $\mathbf{C_H}$ & $\mathbf{C_L}$ = cost functions (proportional to the probability that a successful attack on the existing compromised system occurs),
- $\mathbf{t_N}$ & $\mathbf{t_F}$ = foraging (scanning and system recruitment) times that optimise delivery rate,
- $\mathbf{t_{NL}}$ & $\mathbf{t_{FL}}$= foraging times that maximise net benefit (survival of newly recruited systems into the existing network of compromised hosts) at low "attack" rate ("far" patch affected more than "near" patch. Here systems "deeper" inside network defences are lost first),
- $\mathbf{t_{NH}}$ = optimal time in near patch at high attack rate (far patch no longer confers positive net benefit).

Here, the need for system defence alters foraging behaviour. As more time is required to maintain access to high value systems (defence), less time is available to recruit more systems. Hence, investments in protecting compromised resources lower the amount of capital and time that can be used to scan for and compromise (recruit) more systems.

Where an increased risk of predation affects the market, we see the defending forager acting in a manner to defend systems “closer” to the C & C (Krogoth, 2008).

To maintain a large botnet, the exploitation of systems should be conducted faster, taking valuable data in smaller loads (Kim et al., 2009). If the defence of the system is a goal (the value of each system is sufficient to warrant additional defence), the number of systems in the botnet will decrease due to cost pressure on maintaining the existing systems.

To defend the held resource territory, the criminal attacker needs to maintain access to the compromised host and cover their tracks to avoid detection. There are several activities including the installation of root kits and the deletion of logs that aid the attacker in this goal.

The botnet herder needs to be able to maintain access and control the systems they seek to take advantage of. Even with automation this takes time and resources. Each C & C (command-and-control) server can maintain a limited number of compromised hosts (Pau, 2010). The process of issuing commands, updating systems and maintaining access requires expenditure of time and effort. Each additional system added to a botnet increases the complexity and room for error or detection; hence, the larger the botnet size being maintained, the greater the cost of maintenance. In the case of high-value systems where compromised machines may be manually attacked and rigorously maintained by the cyber-criminal, the costs of maintaining additional hosts can become prohibitive quickly.

All systems contain unknown vulnerabilities that can be exploited by hostile parties. Already compromised systems can be attacked and subsequently compromised again by other attackers who act as predatory

forces against other criminal groups. The difficulty in maintaining access to a compromised host increases when other attackers gain access to vulnerabilities and exploits that are unknown and undefended by both the system owner and any criminal who has already compromised the system.

6.3.5.2 The benefits of a resource

The benefits of running a set of compromised hosts will vary based on the strategy of the cyber-criminal. Some of these strategies include:

- SPAM servers,
- DDoS attack platforms,
- Bitcoin currency mining,
- Bot-For-Rent platforms, and
- Data exfiltration.

The economic viability of each of these platforms varies from large collections of low-value hosts (such as collections of home user machines and anonymous systems) to targeted high-value platforms (including government and defence systems that may contain classified material). The advantages of a model will vary based on the ability of the attacker to maintain that system once it has been acquired.

Criminal territories can be modelled as different ecosystems. These ecosystems vary depending on resource density with the more high-value resources facing far more competition for acquisition from competing criminal elements. To be profitable, low-value systems can be seen to be components of low-resource density ecosystems.

6.3.6 A model of territorial cyber-crime

The process of actively defending a set of resources requires time and effort on the part of the attacker. The more valuable the system is, the

more likely it will be attacked. This results in multiple attacks occurring against high-value targets. The result is active defence from the owner of the system coupled with increased competition from other criminal groups also seeking to exploit high-value resources. The necessity to defend territories (or those systems that have been compromised by the attacker) limits the attacker's ability to recruit (attack and compromise) other systems, exerting a constraint on the territorial gains.

Compromised systems can be lost from the criminal's territory through a combination of defensive strategies from the system owner and predation from competing criminal groups. Competing criminal groups may have differing strategies and reasons for obtaining a system. In some cases, taking over already compromised hosts can be an active strategy designed to reduce the risk associated with the attack. In compromising a system that is already maintained by another criminal group, the predatory criminal can use the existing compromise to cover their own tracks.

Defending a territory requires time and resources. This time taken in actively or passively defending territory is time that cannot be used to expand the territory further. As such, as active defence can be shown to limit access to new systems. The more time required to defend existing territory, the less time there is to acquire new territory. For this reason, there is an upper limit on the size of a territory that can be held with systems requiring a good deal of active maintenance, necessitating more resources to defend systems that are not sought by many others.

The holding of a system requires the exclusion of other parties. In doing this the attacker can more successfully and fully exploit a resource. For the rational criminal (Wright, 2011b), it is worth expending time and effort in these activities to the extent that the additional returns gained

from holding the territory exceed the expenditure of time and energy associated with the territorial behaviour.

To be defendable, a territory comprised of compromised systems must necessarily return a greater net benefit to the attacker or other criminal party then would be available if a non-territorial approach was undertaken.

Where resource density and availability have been lowered through either increased criminal predation on hosts or through improved defence through the system owner, it becomes more likely that the criminal that has already obtained the territory will maintain access to that set of systems. In this instance, defence becomes more likely as a strategy. In these situations, it becomes necessary to consider the resource renewal rate.

In cases where a high depletion rate is coupled with a low renewal rate, it becomes costlier and hence less likely to defend a territory. That is, as large territories become economically costlier to defend, the size of criminal systems such as botnets will become smaller. A high depletion rate will come about when the attacker is unable to maintain access to existing systems. A low renewal rate will come about when systems are more difficult to acquire. In this case, the cost of compromising a system exceeds the amount of time and resources available to the attacker.

Where a high depletion rate is coupled with a high renewal rate, the territory may still be economically defensible. In this instance the territory is likely to come to equilibrium in size at a point where the depletion rate of losing systems and renewal rate of recruiting new systems into the botnet approximately equal one another.

Hypothesis 1: Criminal groups adjust the territory size to the density of the critical resources (such as access to systems, bandwidth or data)

such that the resulting territory contains sufficient benefits to offset the costs of obtaining and maintaining the component systems.

Our first hypothesis can be demonstrated in the variation in botnet sizes and has been modelled by Bensoussan et al. (2010). These researchers proposed that two equilibria exist in criminal botnets, "either (1) the defender group defends at maximum level while the botnet herder exerts an intermediate constant intensity attack effort or (2) the defender group applies an intermediate constant intensity defense effort while the botnet herder attacks at full power".

Hypothesis 2: Variation in territory size occurs because more competitors are attracted to networks that are rich in resources, and such areas are, therefore, costlier to defend per unit host. Kaspersky (Press, 2009b) and iDefense (Danchev, 2010) support this thesis.

6.3.7 Superterritories

The notion of superterritories (Verner, 1977) can be used in modelling criminal behaviour in the creation of large-scale botnets. The notion of selection can be used to compare the fitness of a particular criminal strategy. The use of various types of malware can be modelled as competing against one another as separate criminal groups vie for resources. Individual criminals form either predatory or parasitic strategies against the host they seek to compromise as well as other criminal groups. An individual criminal group can enhance overall fitness either by improving their own absolute performance or by reducing the effectiveness of other criminal groups.

One strategy that can be used to reduce the fitness of competing criminal groups comes from the difficulties seen in maintaining a superterritory. The maintenance of such a territory consisting of botnets of greater than one million compromised hosts reduces the fitness of

other attackers. In maintaining such a large territory, the incumbent criminal restricts access to vital resources and increases the cost of acquisition to other criminal actors (Pau, 2010). This increase in acquisition cost becomes a barrier to entry for new criminal groups. The fitness of those criminals that defend superterritories is only increased where it is still possible to gain profit whilst holding such a territory. The majority use of such territories would be in high-volume low-value transactions such as SPAM.

6.4. Chapter Conclusion

Present crime statistics for cyber-crime more correctly reflect the political state than the actual extent of computer-based crime (Neufeld, 2010). This is a direct consequence of both low reporting and response rates. Many organisations fail to report any computer-based incidents. This can result from a lack of knowledge of the event, a desire to avoid potentially adverse publicity or related consequences, or a failure to meet an economically or legislatively set minimum loss value. These factors undervalue the losses caused by criminal activity.

The overall size of criminal territory results from a compromise between the following factors:

- Acquisition needs,
- Resource maintenance needs,
- Defence costs,
- Predation pressure.

Each of these factors involves an economic cost. Increasing any of those costs results in reduced benefits to the criminal organisation and

hence reduced crime. Modelling the economic costs of cyber-crime allows for the better allocation of resources designed to minimise loss. In making a system more difficult to attack in the first instance, increasing the cost of maintaining access to a compromised system or reducing the amount of time that an attacker can hold access to a compromise system makes it possible to increase the cost to the attacker.

Additionally, it is apparent that predation from both territorial and non-territorial criminals increases the cost associated with cyber-crime. For this reason, high resource-density targets become more expensive to acquire and maintain, leading to smaller territory sizes associated with this criminal strategy. Conversely, low-value targets are more likely to formulate parts of a larger botnet territory.

In this chapter, we have demonstrated that criminal activity can be controlled economically. It is not possible to completely eradicate all crime, but as cyber-criminals seek to maximise returns, any control that introduces costs to the attacker also reduces crime. The result is a balance between competing costs and benefits.

Chapter 7 Conclusion

Security is always a risk exercise and an economic function, one that balances relative risks and rewards of a given system against its alternatives. This dynamic, which, as this thesis has demonstrated, can be represented quantitatively, lies at the crux of the problem. It is not possible to achieve a perfectly secure system. Like all related economic systems, a perfect state of security is one that is infinitely expensive. No state exists where a system could not be compromised or where a slightly greater investment could not make the system slightly more secure. At such a point, small gains start to require exponentially greater investments.

There is no absolute measure of security, but in being able to quantify one system against an alternative, it is possible to better allocate scarce resources and to inhibit the detrimental impact of cyber-crime on online systems. Numerous risk management and information security concerns have come to plague the industry, but solutions exist that do not require expenditure of vast amounts of capital.

Instead, by implementing effective economic incentives and by altering paradigmatic security procedures, organisations can thwart crime by manipulating the relative costs and benefits of criminal acts. One such way involves using one's existing IT infrastructure more effectively. An example would be using NAP and NAC more efficiently. These controls are included freely in modern operating systems, but require knowledge of their existence to be effective. Most systems already have numerous controls that have not even been considered (and in many instances, go

unnoticed). Using these correctly will reduce costs and increase security within an organisation.

Next, a focus on compliance alone can lead to less secure systems. Compliance, security and governance are all related and to some extent interdependent, but only as they focus on the same structures and controls. Better governance does lead to lower risk, but stricter compliance to a set of arbitrary standards does not in and of itself produce a more secure system. For one reason, spending on compliance (as an end unto itself and not a means to an end) involves diverting funds dedicated explicitly to providing security. For example, many compliance regimes miss routers and switches, leaving them in an insecure state that then imperils other sections of the network.

There are a few key aspects of security that are essential and need to be measured. It is essential that we measure compliance, but that we also ensure that compliance is not the goal. For this, continuous monitoring is essential. Automation is an aid, but it does not replace people, it allows them to do more. To achieve this, a firm has a need to develop systems that are designed to be secure and not to simply fulfil a checklist, a systematic tool to ensure that one remembers and executes all the things one should do.

The issue with compliance is that it is backwards facing and this results in a defensive stance that is reactive and difficult to maintain. The result shown in this research is that very few organisations know of the existence of controls such as NAP and NAC, health certificates and the ability to create secure domains. They have not implemented effective monitoring and alerting, let alone good system management. For the most part, organisations are stuck with complex filtering and protocol control as the sole means of securing a network, the use of which leads inevitably to failure without application level controls. The promotion of such

controls is necessary and one path to achieve this end is to demonstrate the economic benefits of such an option.

Patching and system maintenance has been demonstrated in this research to be aligned with audit and not risk. Patching is a test for compliance. While auditors assert that this compliance test is part of good security practice and it is demonstrated that a correctly patched system is a more secure one, what is patched and when is crucial. Whilst some systems are well configured and patched, others are terrible. It all depends on what is audited and whether the employees know about the audit. As systems become more compliant, more is taken from security.

Controls are essential to successful compliance and, more importantly, to the goal of achieving a secure system. They consist of four types: deterrent, preventive, corrective, and detective. The key is implementing and deploying these in a manner that is not simply reactive but is designed to correctly identify and classify risk in the most efficient manner.

As was asserted at the start of the thesis, relative computer security can be measured using six factors (Aycock, 2006):

1. What is the importance of the information or resource being protected?
2. What is the potential impact, if the security is breached?
3. Who is the attacker likely to be?
4. What are the skills and resources available to an attacker?
5. What constraints are imposed by legitimate usage?
6. What resources are available to implement security?

Addressing each of these questions and implementing policy changes or tailor-made controls accordingly is essential to organisational security. The key to achieving this is to ensure that each aspect of the human,

system and software dimensions of security has been considered. In this thesis, we have analysed information security risk using a series of experiments that have been designed to model risk and to return quantitative data. This data can be used to measure the economic impact of security vulnerabilities within an organisation.

The thesis started with the definition of risk and the means to ensure that censored data and evidence can be successfully captured for use. It then moved onto an investigation into the risks associate with software and provided methods that can be used to model and control that risk. The analytic market tools and formula presented can be extended into derivative markets and software survival models. When deployed correctly, these can form the basis of a security risk market place where organisations can select the level of risk they are willing to accept against the costs of mitigating the risk.

This aspect of software security is focused primarily on the development of software, whether in-house or COTS. In both instances, it has been demonstrated that economic methods that return the costs of a risk within a defined confidence interval are available. With this information, a consumer can choose the level of risk they are willing to experience. When extended, we see that a software derivative market functions best where competition allows the consumer to select features or security based on their own needs.

The risks resulting from software bugs and design flaws was followed with an analysis of the system engineering and implementation difficulties that come from the development of increasingly complex systems. In this section, a series of experiments showed that wide scale analysis into the major operating systems and platforms can return actionable intelligence that when deployed effectively will improve compliance. Moreover, it was demonstrated that a one size fits all

approach to security cannot work. From this, we can see that the generalised compliance approach of targeted audits and security controls only leads to activities that are designed to placate management and regulators and not to develop a risk based approach to information security concerns.

As all systems (even those that are to some extent automated) are designed, managed, operated, installed and monitored by people, we next continued the investigations into risk with a series of experiments around the human aspects of security risk. The research in this thesis demonstrated that there are engineering and econometric methodologies that allow organisations to quantify the risk they are facing within defined confidence levels.

When an organisation can quantify the risks to both software and systems, the costs of controls and alternative solutions can be measured and evaluated. The implementation of these controls can then be monitored and the people who are responsible can be held accountable and incentivised to deliver economic returns and not simply meet compliance targets.

In researching the human side of information systems risk, this thesis presents a context that allows organisations to create and implement a comprehensive framework designed to incentivise behaviour designed to minimise risk. In rewarding both teams and individuals for achieving measurable improvements in security and reductions in the total costs over time, we incentivise security over mere compliance.

Finally, we looked at one of the major and growing threat vectors to concern information systems risk, crime. In this section, we demonstrated that all organisations are at risk from criminal groups and extend this research to show how organisations can use the knowledge about the

classes of data and systems that are deployed within their control sphere to reduce the risks they face. In demonstrating that most attackers act rationally, this research demonstrates that as we increase the cost of an attack and hence reduce the payouts from any such activity, an organisation also reduces the incentives for an attacker to target them.

Absolutes do not exist in any economic system, but this is not the goal. The most effective method to controlling risk is to ensure the most economically effective controls are deployed. This research has demonstrated that this is achievable.

7.1. Future research

This thesis presented the initial research efforts in an ongoing set of projects to model risk in information security. This includes the mapping of NAP controls in a Windows environment as well as the impact of awareness sessions on the browsing habits of users when proxy controls are actively and passively measured and reported.

Other efforts will include the creation of models to measure and report on the economic impact of electronic attacks and cyber-crime. The use of multivariate analysis of the data with separate classifications for industry, size, etc. will allow a more detailed level of reporting on collected data. This will extend into an analysis designed to correlate the levels of attacks and statistically analyse the intensity of attacks against each server and host for the collected network traffic and attack data sets as a function of time.

Finally, the quantified risk models created in this research could be applied to the creation of risk instruments for insurance against attacks

and software flaws. These could be used to assign risk to the least costly party and make the true risk more transparent, such that the purchasers of software would not completely avoid risk, but could make an informed decision based on the likely occurrence of an attack for their industry and implementation. Also, the use of controls could be more effectively modelled to allow for the choice of components in a system that best return the desired risk profile to the software user.

Bibliography

Adabinsky, H. (1983). *The Criminal Elite*. Westport, CT: Greenwood Press.

Adams, N. E. (1984). "Optimizing preventive service of software product". *IBM Journal of Research and Development*, *28* (1), 2–14.

Akerlof, G. A. (1970). "The market for 'lemons': quality uncertainty and the market mechanism". *Quarterly Journal of Economics*, *84* (3), 488–500. doi: 10.2307/1879431.

Altmann, M. (1995). "Susceptible-infected-removed epidemic models with dynamic partnerships". *Journal of Mathematical Biology*, *33* (6), 661–675.

Anderson, R. (2001). "Why information security is hard—an economic perspective". Paper presented at the *17th Annual Computer Security Applications Conference*, New Orleans, LA, USA.

Anderson, R., & Moore, T. (2007). "The economics of information security: a survey and open questions". Paper presented at the *Fourth Bi-annual Conference on the Economics of the Software and Interet Industries*, Toulouse, France.

Anderson, R., Moore, T., Nagaraja, S., & Ozment, A. (2007). “Incentives and information security”. In N. Nisan, T. Roughgarden, E. Tardos & V. V. Vazirani (Eds.), *Algorithmic Game Theory*. Cambridge: Cambridge University Press, pp. 633–649

Archer, M. S. (Ed.), Tritter, J. Q. (Ed.). (2000). *Rational Choice Theory: Resisting Colonization* “*Homo economicus, Homo sociologicus* and *Homo sentiens*” Archer, M. S., pp. 36-56: Routledge.

Armstrong, M. (1994). *Performance Management*. London: Kogan Page.

Arora, A., Krishnan, R., Telang, R., & Yang, Y. (2005). “An empirical analysis of vendor response to disclosure policy”. Paper presented at the *Fourth Annual Workshop on Economics of Information Security* (WEIS05), Harvard University.

Arora, A., Nandkumar, A., & Telang, R. (2006). “Does information security attack frequency increase with vulnerability disclosure? An empirical Information Systems Frontiers analysis”. *Information Systems Frontiers*, *8* (5), 350–362.

Arora, A., & Telang, R. (2005). “Economics of software vulnerability disclosure”. *IEEE Security and Privacy*, *3* (1), 20–22.

Arora, A., Telang, R., & Xu, H. (2004). “Optimal Time Disclosure of Software Vulnerabilities”. Paper presented at the *Conference on Information Systems and Technology*, Denver, CO.

Arora, A., Telang, R., & Xu, H. (2008). “Optimal Policy for software vulnerability disclosure. *Management Science*, *54* (4), 642–646.

August, T., & Tunca, T. I. (2006). “Network software security and user incentives”. *Management Science*, *52* (11), 1703–1720.

Aycock, J. (2006). “Computer viruses and malware”. *Advances in Information Security, 22.*

Bacon, D. F., Chen, Y., Parkes, D., & Rao, M. (2009). “A market-based approach to software evolution”. In *Proceedings of the 24th ACM SIGPLAN conference Companion On Object-Oriented Programming Systems Languages and Applications*. New York, NY, USA, pp. 973-980

Badonnel, R., State, R., Chrisment, I., & Festor, O. (2007). “A Management platform for tracking cyber predators in peer-to-peer networks”. Paper presented at the *Second International Conference on Internet Monitoring and Protection*, San Jose, CA.

Bamber, G. J., Shadur, M. A., & Howell, F. (1992). “The international transferability of Japanese management strategies: An Australian perspective”. *Employee Relations, 14* (3), 3–19.

Banerjee, D., Jones, T. W., & Cronan, T. P. (1996). “The association of demographic variables and ethical behaviour of information system personnel”. *Industrial Management & Data Systems*, *96* (3), 3–10.

Bang, H.; & Tsiatis, A. A. (2000) "Estimating medical costs with censored data" Biometrika 87 (2): 329-343 doi:10.1093/biomet/87.2.329

Baskerville, R. (1993). “Information systems security design methods: implications for information systems development”. *ACM Computing Surveys*, *24* (4), 375–414.

Basu, S. (1977). “Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient markets hypothesis”. *Journal of Finance*, *32*, 663–682.

Bayes, T. (1763). “An essay towards solving a problem in the doctrine of chances”. *Philosophical Transactions of the Royal Society*, *53*, 370–418.

Beach, J. R., & Bonewell, M. L. (1993). “Setting-up a successful software vendor evaluation/qualification process for ‘off-the-shelf’ commercial software used in medical devices”. In *Proceedings of Sixth Annual IEEE Symposium on Computer-Based Medical Systems*. Ann Arbor, Michigan, pp. 284-288

Bednar, P. M., Katos, V., & Hennell, C. (2008). “Cyber-crime investigations: complex collaborative decision making”. Paper presented at the *Digital Forensics and Incident Analysis, WDFIA ’08. Third International Annual Workshop*. Malaga, Spain

Bednarski, G. M., & Branson, J. (2004). *Information warfare: understanding network threats through honeypot deployment*: Carnegie Mellon University.

Bellovin, S. (2008). “Security by checklist”. *Security & Privacy*, 6 (2), 88.

Bellovin, S. (2009). *Security Analysis 1 COMS W4187*. Columbia USA.

Ben-Itzhak, Y. (2009). "Organised cybercrime and payment cards". *Card Technology Today*, *21* (2), 10–11.

Bender, M. (2002) "The Sarbanes-Oxley Act of 2002: with analysis", LexisNexis

Bensoussan, A., Kantarcioglu, M., & Hoe, S. (2010). "A game-theoretical approach for finding optimal strategies in a botnet defense model". In *Proceedings of the First International Conference on Decision and Game Theory for Security*, Berlin, Germany. pp. 135-148

Benveniste, A. Jacod, J. (1973). "Systèmes de Lévy des processus de Markov". *Invent. Math Mathematical Reviews*, *21*, 183–198.

Bernstein, Peter L. (1992) *Against the Gods: The Remarkable Story of Risk.* Wiley (August 31, 1992).

Bishop, M., & Frincke, D. A. (2005). "Teaching secure programming". *IEEE Security & Privacy*, *3* (5), 54–56.

Blakley, B. (2002). "The measure of information security is dollars". Paper presented at the *First Workshop on Economics and Information Security*, May 16–17, 2002. University of California, Berkeley

Brache, A. P., & Rummler, G. A. (1995). *Improving Performance* (2nd ed.). San Francisco: Jossey Bass.

Bradley, T. “Zero day exploits, holy grail of the malicious hacker”. *About.com Guide.* Date accessed: 15 April 2012

Brémaud, P. (1981). *Point processes and queues: Martingale dynamics*. New York: Springer.

Brito, D. L., Sheshinski, E., & Intriligator, M. D. (1991). “Externalities and compulsory vaccinations”. *Journal of Public Economics*, *45*, 69–90.

Broadhurst, R. G. (2005). “Measures to combat computer-related crime”. Paper presented at the International Cooperation in Cyber-crime Research. In *Proceedings 11th UN Congress on Crime Prevention and Criminal Justice*, Workshop 6, Bangkok. pp. 1-12

Broadhurst, R. G., & Grabosky, P. N. (2005). *Cyber-crime: the challenge in Asia*. Hong Kong: Hong Kong University Press.

Broersma, M. (2005). “Linux servers safer than ever”. *Techworld*. Published 20 January 2005

Brooks, F. (1995). *The mythical man-month*. Addison-Wesley. Boston, MA, USA

Brown, J. L. (1964). “The evolution of diversity in avian territorial systems”. *Wilson Bulletin* (76), 160–169.

Brown, W., & Walsh, J. (1994). “Managing pay in Britain”. In O. B. Business. (Ed.), *Personnel management: a comprehensive guide to theory and practice in Britain.* (2nd ed.). Publisher Wiley-Blackwell, Oxford.

Camp, L. J., & Wolfram, C. (2000). “Pricing security”. Paper presented at the *CERT Information Survivability Workshop*, October 24–26. Boston, Massachusetts USA

Campbell, K., Gordon, L. A., Loeb, M. P., & Zhou, L. (2003). “The economic cost of publicly announced information security breaches: empirical evidence from the stock market”. *Journal of Computer Security*, *11*, 431.

Campodonico, S. (1994). “A Bayesian analysis of the logarithmic-poisson execution time model based on expert opinion and failure data”. *IEEE Transactions on Software Engineering*, *20*, 677–683.

Carman, D. W., Dolinsky, A. A., Lyu, M. R., & Yu, J. S. (1995). “Software Reliability Engineering Study of a Large-Scale Telecommunications System”. In *Proceedings of the Sixth International Symposium on Software Reliability Engineering* Toulouse, France, pp. 350-359

Carroll, L. (1871). *Through the looking-glass and what Alice found there*: Macmillan.

Carter, P., & Jackson, N. (2000). *Rethinking organisational behaviour.* UK: Financial Times and Prentice Hall.

Cavusoglu, H., Cavusoglu, H., & Zhang, J. (2006). “Economics of Security patch management”. Paper presented at the *Fifth Workshop on the Economics of Information Security (WEIS)*. University of Cambridge, England

Chakrabarty, A. Guo, X. (2007). *A note on optimal stopping times with filtration expansion*: University of California, Berkeley. Chapter in Stochastic Analysis and its Application to Mathematical Finance, World Scientific Publishers, 2011 pp. 19-38

Chao-Hsi Yeh; Grad. Inst. of Inf. & Comput. Educ., Nat. Kaohsiung Normal Univ., Kaohsiung ; Chung-Huang Yang (2008) *Design and implementation of honeypot systems based on open-source software.* Intelligence and Security Informatics, 2008. ISI 2008. IEEE International Conference

Cheng, P., Rohatgi, P., Keser, C., Karger, P. A., Wagner, G. M., Reninger, A. S. (2007). "Fuzzy multi-level security: an experiment on quantified risk-adaptive access control". In *Security &Privacy* 2007. ACM: Oakland, California, USA; 222–230.

Choi, J. P., Fershtman, C., & Gandal, N. (2005). "Internet security, vulnerability disclosure, and software provision". Paper presented at the *Fourth Workshop on the Economics of Information Security*. Kennedy School of Government Harvard University

Christopher, A. (2003). "The human firewall". http://cio.co.nz/cio.nsf/0/CD50373FD1A06B D3CC256DCD00015C68, Last accessed 15 March 2011

Ciocchetti, C. A. (2010). The eavesdropping employer: a twenty-first century framework for employee monitoring. Future of Privacy Forum.

Clarke, R., & Cornish, D. (Eds.). (1985). *Modelling offender's decisions: A framework for research and policy*: Chicago: University of Chicago Press.

Cobb, C. W., & Douglas, P. H. (1928). "A theory of production". *American Economic Review, 18* (1), 139–165.

Coe, K. (2003). "Closing the security gap: Data protection initiatives should include employee training". *HR Magazine, 48* (8), 95ff.

Cohen, J. (2006). "Best kept secrets of peer code review: Modern approach. practical advice". Samrtbearsoftware (2006)

Cohen, P. S. (1976). *Rational conduct and social life. Rationality and the Social Sciences: Contributions to the Philosophy and Methodology of the Social Sciences*. In S. I. Benn and G. W. Mortimore (Eds.), *Rationality and the Social Sciences: Contributtions to the Philosophy and Methodology of the Social Sciences,* London: Routledge and Kegan Paul, pp. 132-154]

Cohen, Fred (1997) ""Pen.Testing"?" http://all.net/journal/netsec/1997-08.html

Cohen, Fred (1998) "Red Teaming and Other Agressive Auditing Techniques" http://all.net/journal/netsec/1998-03.html

Connell, C. (2003). “It’s Not About Lines of Code”. Retrieved from http://www.developer.com/java/other/article.php/988641, Last accessed 12 Dec 2011

Corcuera, J. M., Imkeller, P., Kohatsu-Higa, A. and Nualart, D. (2004). “Additional utility of insiders with imperfect dynamic information”. *Finance and Stochastics, 8*, 437–450.

Curtis V, Kanki B, Cousens S et al. (2001) Evidence for behaviour change following a hygiene promotion programme in West Africa. Bulletin of the World Health Organization 79, Pp. 518–526

Danchev, D. (2010). “Study finds the average price for renting a botnet”. *Zero Day* Retrieved from http://www.zdnet.com/blog/security/study-finds-the-average-price-for-renting-a-botnet/6528 Date accessed: 25 Jan 2012

Davies, I. K. (1994). “Process re-design for enhanced human performance”. *Performance Improvement Quarterly, 7* (3), 103–113.

Davis, D. D., & Holt, C. A. (1993). *Experimental economics*. Princeton: Princeton University Press.

Dellacherie, C. Meyer., P. A. (1982). *Probabilities and potential*. North-Holland, Amsterdam.

Dempster, A. P. Laird, N. M. and Rubin, D. B. Dempster A. P, Laird N. M., Rubin D. B (1977). “Maximum likelihood from incomplete

data via the EM algorithm". *Journal of the Royal Statistical Society, 39* pp. 1-38

Denzin, N. K., & Lincoln, Y. S. (1998). *Collecting and interpreting qualitative materials*. Thousand Oaks, CA.: Sage Publications.

Devanbu, P. T (2000). "Software engineering for security: a roadmap". In *Proceedings of the Conference on The Future of Software Engineering*, Limerick, Ireland. pp. 227-239

Devost, M. G. (1996). *Hackers as a national resource. Information warfare–cyberterrorism: Protecting your personal security in the electronic age*. New York: Thunder's Mouth Press.

Dhillon, G. (2001). *Information security management: global challenges in the new millennium*: Idea Group Pub. Hershey, Pennsylvania

Dhillon, G., & Backhouse, J. (2001). "Current directions in IS security research: towards socio-organizational perspectives". *Information Systems Journal*, *11*, 127–153.

Dijstra, Edsger W. (1976). A Discipline of Programming. Englewood Cliffs, NJ: Prentice Hall

Dodson, Bryan & Nolan, Dennis (2005 Ed) "The Reliability Engineering Handbook" Quality Publishing.

Donald, D. (2006). *Economic Foundations of Law and Organisation*: Cambridge University Press.

DSD. (2012). 2012 "Australian Government Information Security Manual, Principles", retrieved from

http://www.dsd.gov.au/publications/Information_Security_Manual_ 2012_Principles.pdf. Last accessed 05 May 2011

DShield. (2006–2010), from http://www.dshield.org. Last accessed 25 May 2014

Durtschi, C., Hillison, W., & Pacini, C. (2002). "Web-based contracts: You could be burned!" *Journal of Corporate Accounting & Finance, 13* (5), 11–18.

Einstadter, W., & Stuart, H. (1995). *Criminological Theory*. Fort Worth: Harcourt Brace.

Elberzhager, F., Klaus, A., & Jawurek, M. (2009). "Software inspections using guided checklists to ensure security goals". Paper presented at the *International Conference on Availability, Reliability and Security*. Fukuoka Institute of Technology, Japan 16-19 March 2009

Elliott, R. J., Jeanblanc, M. and Yor, M. (2000). "On models of default risk". *Journal of Mathematical Finance, 10*, 1799–1195. pp. 179-195

Fisk, M. (2002). "Causes & remedies for social acceptance of network insecurity". Paper presented at the *First Workshop on Economics and Information Security*, University of California, Berkeley, May 16–17, 2002.

Fitz-enz, J. (1997). It's costly to lose good employees". *Workforce, 75*, 50-51.

Fowler, C. A., & Nesbit, R. F. (1995). “Tactical deception in air-land warfare”. *Journal of Electronic Defense, volume? 18(6)* pp. 37-44 & 76-79.

Friedman, M. (1953). *The Methodology of Positive Economics*. Chicago: Chicago and London: Chicago University Press.

Gambetta, D. (1988). “Fragments of an economic theory of the mafia”. *Archives Européennes de Sociologie, 24*. XXIX, 1, 127-145

Gambetta, D. (1991). *The origins of the mafias*. Cambridge: Mimeo.

Gambetta, D. (1993). *The Sicilian mafia: The business of protection*. London: Harvard University Press.

Gambetta, D. (Ed.). (2000). *Mafia: the price of distrust*. New York: Basil Blackwell.

Gawande, A. (2007). The checklist: Annals of medicine. *The New Yorker*.

Gawande, A. (2009). *The checklist manifesto: How to get things right*. Macmillan. New York, NY, USA]

Geoffard, P. Y., & Philipson, T. (1996). “Rational epidemics and their public control”. *International Economic Review, 37* (3), 603–624.

Ghosh D, Lin D.Y. (2000). Nonparametric analysis of recurrent events and death. Biometrics, 56:554-562.

Ghoshal, S., & Bartlett, C. A. (1995). Changing the role of top management: beyond structure to processes. *Harvard Business Review*, 86-96.

Gordon, L. A., & Loeb, M. P. (2002). "The economics of information security investment". *ACM Transactions on Information and System Security, 5* (4), 438–457.

Gordon, S., & Ford, R. (2002). "Cyberterrorism?", from http://www.symantec.com/avcenter/ reference/cyberterrorism.pdf. Last accessed 02 Febuary 2014

Grabosky, P., & Broadhurst, R. G. (Eds.). (2005). *The future of cyber-crime in Asia*. Hong Kong: The University of Hong Kong Press.

Grandell, J. (1991). *Aspects of risk theory*. New York: Springer.

Grembergen, W. V. (Ed.). (2004). *Strategies for information technology governance*. Idea Group Publishing.

Guo, X., Jarrow, R., and Zeng, Y. (2005). *Modeling the recovery rate in a reduced form model*. Preprint, Cornell Univ.

Hahn, R. W., & Layne-Farrar, A. (2006–2007). The Law and Economics of Software Security. April 2006 AEI-Brookings Joint Center Working Paper No. 06-08

Halderman, J. (2010). "To Strengthen Security, Change Developers' Incentives". *IEEE Security and Privacy, 8* (2), 79–82.

Hale, P (2002). "Microsoft now serious about bugs, says Ballmer". *The Inquirer*, Last Accessed 01 June 2014. http://www.theinquirer.net/inquirer/news/1007300/microsoft-now-serious-about-bugs-says-ballmer .

Hales, B., & Pronovost, P. (2006). “The checklist—a tool for error management and performance improvement”. *Journal of Critical Care, 21* (3), 231–235.

Hawkins, S., Yen, D. C., & Chou, D. C. (2000). Awareness and challenges of Internet security. *Information Management & Computer Security, 8*(3), 131-143.

He, S. W., Wang, J. G., & Yan, J. A. (1992). *Semimartingale theory and stochastic calculus*. Beijing: Science Press.

Hechter, M., & Kanazawa, S. (1997). “Sociological rational choice theory”. *Annual Review of Sociology, 23* (1), 191–214. doi: 10.1146/annurev.soc.23.1.191.

Hein, D. (2011). *Sicherheitsaspekte in der Softwareentwicklung [Security aspects in software development]*. WS 11/12. Graz - University Of Technology.

Herzberg, F., Mausner, B., & Snyderman, B. (1959). The motivation to work (2nd ed.). New York: John Wiley and Sons.

Hind, P. (2004). Give it away, take my security please… (At the Coal Face). *CIO Magazine*.

Hoevemeyer, V. A. (1989). “Performance-based compensation: miracle or warfare?” *Personnel Journal, 68* (7), 64.

Hofmeyr, S. A., Moore, T., Forrest, S., Edwards, B., & Stelle, G. (2011). “Modeling Internet-scale policies for cleaning up malware”. Paper presented at the WEIS June 14-15, 2011, Fairfax, VA

http://crd.lbl.gov/assets/pubs_presos/CDS/FTG/Papers/2011/weis2011-cleaning-malware.pdf last accessed 15 March 2013

Hoo, K., & Soo, J. (2000). *How much is enough? A risk-management approach to computer security.* (Unpublished doctoral dissertation). Stanford University.

Hsu, Chiu-Hsieh; Taylor, Jeremy M. & Hu, G.Chengcheng (2015) "Analysis of accelerated failure time data with dependent censoring using auxiliary variables via nonparametric multiple imputation", Statistics in Medicine, 2015, 34, 19, 2768, Wiley Online Library (http://onlinelibrary.wiley.com/doi/10.1002/sim.6534/full)

Ikeda, N. Watanabe, S. (1962). "On some relations between the harmonic measure and the Lévy measure for a certain class of Markov processes". *Journal of Mathematics,* Kyoto University, *2*, 79–95.

Ivancevich, J. M., & Matteson, M. T. (1999). *Organisational Behaviour and Management*. Singapore: McGraw Hill.

Jackson, N. (1992). "Training needs: An objective science?" In N. Jackson. (Ed.), *In Training for What? Labour Perspectives on Job Training*. Toronto: Our Schools/Our Selves Education Foundation. pp. 76-83

Jacod, J. (1975). "Multivariate point processes: Predictable projection, Radon–Nikodým derivatives, representation of martingales". *Z. Wahrsch. Verw. Gebiete, 31*, 235–253.

Jaisingh, J., & Li, Q. (2005). “The optimal time to disclose software vulnerability: Incentive and commitment”. Working paper. Hong Kong University of Science and Technology, Hong Kong.

JASON Report 2004: JASON. (2004) “Horizontal integration: broader access models for realizing information dominance”. Technical Report JSR-04-132, MITRE Corporation. http://www.fas.org/irp/agency/dod/jason/classpol.pdf Last Accessed 31 Jan 2013

Jaziar, R. (2007). “Understanding Hidden Information Security Threats: The Vulnerability Black Market”. Paper presented at the *40th Annual Hawaii International Conference on System Sciences*, Hawaii.

Jeanblanc, M., & Valchev, S. (2005). “Partial information and hazard process”. *International. Journal of Theoretical and Applied Finance, 8*, 807–838.

Kaaniche, K., & Kanoun, K. (1996). “Reliability of a telecommunications system”. In *Proceedings of the Seventh International Symposium Software Reliability Engineering*. Crowne Plaza Hotel, White Plains, NY Oct. 30 1996-Nov. 2 1996, 1996 Article pp. 207-212

Kahneman, D. and Tversky, A. (1984). “Choices, Values, and Frames”. *American Psychologist*. *39* (4): 341–350. doi:10.1037/0003-066x.39.4.341.

Kannan, K., & Telang, R. (2004). “Market for Software vulnerabilities? Think again”. *Management Science, 58* (5), 726–740.

Katz, M. L., & Shapiro, C. (1985). “Network externalities, competition, and compatibility”. *The American Economic Review, 75*, 424.

Kay, R. (1977). “Proportional hazard regression models and the analysis of censored survival data”. *Journal of the Royal Statistical Society. Series C (Applied Statistics), 26* (3), 227–237.

Kayser, Olivier & Budinich, Valeria (2015) " When markets fail" Scaling up Business Solutions to Social Problems; in Scaling up Business Solutions to Social Problems, pp 92-103, Palgrave Macmillan UK

Keong, Tan Hiap (2004) “Risk Analysis Methodologies” http://pachome1.pacific.net.sg/~thk/risk.html (Last Viewed 27th December 2005)

Khoshgoftaar, T. M., Allen, E. B., Kalaichelvan, K. S., & Goel, N. (1996). “Early quality prediction: A case study in telecommunications”. *IEEE Transactions on Software Engineering, 13* (1), 65–71.

Kim, D., Lee, T., In, H. P. & Jeong, H.C. (2009). “Bonet Damage Propagation Estimation Model”. Paper presented at the *First International Conference on Internet (ICONI)*. Nusa Dua (Bali), Indonesia 17-21 December, Retrieved from

http://embedded.korea.ac.kr/esel/paper/ international/2009/1200910.pdf, Last accessed 01 Jan 2013

Kolstad, C. D., & Mathiesen, L. (1991). "Computing Cournot-Nash equilibria". *Operations Research, 39*, 739–748.

Kovacich, G. L. (2003). *The information systems security officer's guide: establishing and managing an information protection program*. Elsevier Science. http://books.google.com.au/books/about/The_Information_Systems_Security_Officer.html?id=6LlBZ6kzoegC&redir_esc=y + link of contents https://www.elsevier.com/books/the-information-systems-security-officers-guide/kovacich/978-0-7506-7656-4

Kramer, R., McGraw, P., & Schuler, R. (1997). *Human resource management in Australia* (3rd ed.). South Melbourne: Longman.

Krogoth. (2008). *Botnet constuction, control and concealment.* Unpublished MSc thesis. Retrieved from https://www.botnets.fr/index.php/Botnet_construction,_control_and _concealment. Last Accessed 30 March 2013

Kunreuther, H., & Heal, C. (2005). Interdependent security: A General model. *Journal of Risk and Uncertainty, 26*, 231-249.

Kunreuther, H., Meyer, R., Zeckhauser, R., Slovic, P., Schwartz, B., Schade, C., Hogarth, R. (2002). High stakes decision making: Normative, descriptive and prescriptive considerations. *Marketing Letters, 13*(3), 256-268.

Kuo, L., & Yang, T. Y. (1996). “Bayesian computation for nonhomogeneous poisson processes in software reliability”. *Journal of the American Statistical Association, 91*. pp. 763-773

Kurz, M., & Hart, S. (1982). “Pareto-optimal Nash equilibria are competitive in a repeated economy”. *Journal of Economic Theory, 28*, 320–346.

Lane, D. (2004). *Foundations of HRM, performance and compensation management.* Management Course Notes, University of SA.

Lansbury, R., Braithwaite, J., & Westbrook, J. (Eds.). (1995). *Goal-directed approaches to performance appraisal*. Melbourne, Vic.: Pitman Publishers.

Lee, C. (1996). Performance appraisal: can we ‘manage’ away the curse? *Training*, 44, 46-48, 50, 53, 55, 57, 59.

Levendel, Y. (1990). “Reliability Analysis of Large Software Systems: Defects Data Modeling”. *IEEE Transactions Software Engineering, 16* (2), 141–152.

Li, Z., Liao, Q., & Striegel, A. (2009). “Botnet economics: Uncertainty matters”. *Managing Information Risk and the Economics of Security* (pp. 245–267). Springer US.

Lin, D. Y., Feuer, E. J., Etzioni, R., & Wax, Y. (1997). “Estimating medical costs from incomplete follow-up data”. *Biometrics, 53*, 419–434.

Lin, J.-C., Chen, J.-M., Chen, C.-C., & Chien, Y.-S. (2009). “A game theoretic approach to decision and analysis in strategies of attack and defense”. In *Proceedings of the 2009 Third IEEE International Conference on Secure Software Integration and Reliability Improvement*. Shanghai, China - July 8-10 2009, pp. 75-81

Linde, R.R., "Operating System Penetration," Proceedings of the National Computer Conference, Vol. 44, AFIPS Press, Montvale, N.J., 1975

Locke, E. A., & Latham, G. P. (1990). “Work motivation and satisfaction: Light at the end of the tunnel”. *Psychological Science, 1*, 240–246.

Longley, A. D., & Kwok, L. F. (1994). *Security modeling for organisations*. Paper presented at the *ACM Conference on Computers and Communications Security*. Fairfax, VA, USA - November 02-04 1994

Lui, D., Li, N., Wang, X., Camp, L. J. (2011). “Security risk management using incentives”. *IEEE Security & Privacy*, *9* (6), 20–28.

Lyman, M. D., & Potter, G. W. (1997). *Organized crime*. New Jersey: Prentice Hall.

Mann, C. (2002). Homeland insecurity. *The Atlantic Monthly*. Retrieved from http://www.docstoc.com/docs/111584185/Homeland-Insecurity

Marti, K. (2008). “Computation of probabilities of survival/failure of technical, economic systems/structures by means of piecewise linearization of the performance function”. *Structural and Multidisciplinary Optimization, 35* (3), 225–244.

Martin, J. (1973) “Security, Accuracy and Privacy in Computer Systems”, Prentice Hall USA

McCarthey, J. (2001). “Risk management: Plan for people, not just system”. *CIO Magazine*. Nov. 15, 2001.

McGrew, R. (2006) “*Experiences with Honeypot Systems: Development, Deployment, and Analysis.* System Sciences, 2006. HICSS '06. Proceedings of the 39th Annual Hawaii International Conference on System Sciences (Volume:9) Mississippi State University

Mead, R. (1998). *International Management, cross-cultural dimensions* (2nd ed.). Blackwell Publishing.

Mills, H. D. (Ed.). (1971). *Top-down programming in large systems. Debugging techniques in large systems*. Englewoods Cliffs, NJ: Prentice-Hall.

Mitnick, K. D., & Simon, W. L. (2002). *The art of deception: Controlling the human element of security*: John Wiley & Sons, Inc. United States

Molloy, I., Cheng, P. C., Rohatgi, P. (2008) “Trading in risk: Using markets to improve access control”. In *Proceedings of New*

Security Paradigms Workshop (NSPW'08). ACM: lake Tahoe, California, USA, 2008; 1–19.

. Moore, Andrew P., Ellison, Robert J. & Linger Richard C. (2001) "Attack Modeling for Information Security and Survivability", Carnegie Mellon University. The Software Engineering Institute US

Moore, D., Paxson, V., Savage, S., Shannon, C., Staniford, S., & Weaver, N. (2003). *The spread of the Sapphire/Slammmer worm*. Working paper. CAIDA. Berkeley, CA.

Morash, M. (Ed.). (1984). *Organized crime*. California: Sage Publications.

Mougeot, M. & Naegelen, F. (2009). "Adverse Selection, Moral Hazard, and Outlier Payment Policy", *Journal of Risk and Insurance*, Volume 76-1, pp. 177-195, Blackwell Publishing Inc. USA.

Munson, J. C., & Khoshgoftaar, T. M. (1992). "The detection of fault-prone programs". *IEEE Transactions on Software Engineering, 18* (5), 423–433.

Murphy, R. P. (2009). *The politically incorrect guide to the Great Depression and the New Deal*: Regenery Publishing. Washington, DC

Myagmar, Suvda, Lee Adam J. & Yurcik, William (2005) "Threat Modeling as a Basis for Security Requirements", National Center for Supercomputing Applications (NCSA)

Myers, I. B., & McCaulley, M. H. (1985). *Manual: A guide to the development and use of the Myers-Briggs Type Indicator* (2nd ed.). Palo Alto, CA: Consulting Psychological Press.

Maish, D. (2013). "Tales from the Cryptozoologicon: BUNYIP". *Scientific American* October 12, 2013, V. 91.

Nankervis, A., & Leece, P. (1997). "Performance appraisal: Two steps forward, one step back? ". *Asia Pacific Journal of Human Resources, 35* (2), 80–92.

Neufeld, D. J. (2010). "Understanding cybercrime". Paper presented at the *43rd Hawaii International Conference on System Sciences*, Hawaii.

Newman, M. E. J., Strogatz, S. H., & Watts, D. J. (2001). "Random graphs with arbitrary degree distributions and their applications". *Physics Review Editorial, 64*. E, 64 (2):206118

Ni, Q., Bertino, E., Lobo, J. (2010). "Risk-based access control systems built on fuzzy inferences". In ASIACCS, Feng D, Basin DA, Liu P (eds.) ACM; 250–260. In: ASIACCS 2010, pp. 250-260. ACM, New York

Nicastro, F. (2005). "Network security tactics. Step-by-step guide: How to deploy a successful patch". Retrieved from http://www.searchsecurity.techtarget.com/. date last accessed: 04 Jan 2013

Nicholson, F. (1968). “Price-earnings ratios in relation to investment results”. *Financial Analysts Journal* (Jan/Feb), 105–109.

Nisan, N., Roughgarden, T., Tardos, E., & Vazirani, V. (Eds.). (2007a). *Algorithmic Game Theory*. Cambridge University Press.

NIST, National Institute of Standards and Technology. Building an Information Technology Security Awareness and Training Program. Wilson, M. and Hash, J. Computer Security Division Information Technology Laboratory. October 2003. Retrieved from:http://csrc.nist.gov/publications/nistpubs/800-50/NIST-SP800-50.pdf

O’Brien, J., A. (1999). *Management Information systems: Managing information technology in the Internetworked enterprise* (4th ed.). Irwin McGraw-Hill.

O’Neill, G., & Kramar, R. (1995). *Australian Human Resources Management*. Melbourne: Pitman.

Ozment, A., & Schechter, S. E. (2006). “Bootstrapping the adoption of Internet security protocols”. Paper presented at the *Fifth Workshop on the Economics of Information Security*, Cambridge.

Parameswaran, M., Rui, H., & Sayin, S. (2010). “A game theoretic model and empirical analysis of spammer strategies”. Paper presented at the *CEAS 2010: Collaboration, Electronic messaging, Anti-Abuse and Spam Conference.* Redmond, Washington, July 13-14 2010

Pau, L.-F. (2010). Botnet economics and devising defence schemes from attackers' own reward processes. MPRA Paper from University Library of Munich, Germany.

Pauna, A. (2014). RASSH - Reinforced adaptive SSH honeypot, Military Technical Academy, Faculty of Military Electronic and Information Systems, Bucharest, Romania May 29-31 2014 pp. 1-6

Peisert, S., & Bishop, M. (2007). *How to design computer security experiments*. Paper presented at the WG 11.8 International Federation of Information Processing, Boston.

Perrow, C. (1984). *Normal accidents: Living with high-risk technologies*. New York: Basic Books.

Pheh, "edited by unknown authors" . (2008). *RBN as a Business Network: Clarifying the guesswork of criminal activity*. The ShadowServer Foundation.

Pillai, R. K. G., & Kumar, P. R. (2007). "Simulation of human criminal behavior using clustering algorithm". Paper presented at the *ICCIMA* 2007. International Conference on Computational Intelligence and Multimedia Applications - 13-15 December, 2007, Sivakasi, Tamil Nadv, India

Press, T. A. (2009a). Famous hacker Kevin Mitnick gets hacked. *CBS*. Retrieved from http://www.cbsnews.com/2100-205_162-540191.html

Press, T. A. (2009). “Kaspersky reveals price list for botnet attacks”. Retrieved from http://www.computerweekly.com/news/1280090242/Kaspersky-reveals-price-list-for-botnet-attacks, date last accessed: 14 December 2013

Radianti, J., & Gonzalez, J. J. (2006). “Toward a dynamic modeling of the vulnerability black market”. Paper presented at the *Workshop on the Economics of Securing the Information Infrastructure*. Arlington, VA. October 23-24, 2006

Radvanovsky, B., Wright, C. S., Brodsky, J., Weiss, J., & Harley, D. (2012). *Handbook on SCADA/Control Systems Security*. Boca Raton, Florida: Taylor & Francis Group.

Revuz, D., & Yor, M. (1999). *Continuous Martingale and Brownian motion*. Berlin: Springer.

Richards, J. R. (1999). *Transnational criminal organisations, cybercrime, and money laundering*. Boca Raton, Florida: CRC Press LLC.

Roese, N. J., & Olson, J. M. (2007). “Better, stronger, faster: Self-serving judgment, affect regulation, and the optimal vigilance hypothesis”. *Perspectives on Psychological Science, 2*, 124–141.

Romeo, J. (2002). “Keeping your network safe, HR must protect sensitive data from internal and external security threats”. *HR Magazine, 47* (12). P 42

Rosenberg, B., Reid, K., & Lanstein, R. (1985). “Persuasive evidence of market inefficiency”. *Journal of Portfolio Management, 13*, 9–17.

RTI. (2002). The economic impacts of inadequate infrastructure for software testing. A report prepared by RTI for NIST. Retrieved from http://www.nist.gov/director/planning/upload/report02-3.pdf

Samuelson, P. (1972). “Proof that properly anticipated prices fluctuate randomly”. *Industrial Management Review, 6* (2), 41–49.

SANS, (2007) "20 Critical Security Controls, Twenty Critical Security Controls for Effective Cyber Defense: Consensus Audit Guidelines" (http://www.sans.org/critical-security-controls/)

Savage, M. (2003). “Hiring hackers: A heated debate”. *Techweb*. Retrieved from http://www.techweb.com/wire/story/TWB20030416S0003. 16th Apr 2003

Schalb, M. (2007). “Exploit derivatives & national security”. *Yale Journal of Law and Technology: Vol. 9, 9*(5). Vol. 9: Iss. 1, Article 5

Schechter, S. E. (2004). *Computer security strength & risk: A quantitative approach.* Unpublished PhD Dissertation, Harvard University, Cambridge, Massachusetts.

Schneier, B. (2002). “No, we don’t spend enough!” Paper presented at the *First Workshop on Economics and Information Security*. University of California, Berkeley. May 16-17, 2002

Schneier, B. (2007). “A security market for lemons”. Retrieved from http://www.schneier.com/blog/archives/2007/04/a_security_mark.html Last accessed: 22 November 2013

Scott, M. D. (2007). “Tort liability for vendors of insecure software: has the time finally come? ” *Maryland Law Review, 67,* 425.

Sestoft, P. (2008). *Systematic software testing*. IT University of Copenhagen. Denmark.

Shannon, C., & Moore, D. (2004). “The spread of the witty worm”. *IEEE Security Privacy, 2* (4), 46–50.

Shawgo, J., Whitney, N., & Faber, S. (2005) CIS Windows XP Professional Benchmark v.2.0.1. [Windows XP Professional Operating System Legacy, Enterprise,and pecialized Security Benchmark Consensus Baseline Security Settings. V 2.01 August, 2005]

Sheard, J., & Carbone, A. (2004). “From informal to formal: creating the Australasian computing education community”. Paper presented at the *6th Australasian Computing Education Conference (ACE2004)*, Dunedin.

Skyrms, B. (2004). *The stag hunt and the evolution of social structure*: Cambridge University Press.

Stolpe, M. (2000). “Protection against software piracy: a study of technology adoption for the enforcement of intellectual property

rights". *Economics of Innovation and New Technology, 9* (1), 25–52.

Stone, R. (2002). *Human resource management.* (4th ed.). Brisbane: Wiley.

Strawderman, R. (2000) "Estimating the Mean of an Increasing Stochastic Process at a Censored Stopping Time" JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION 95(452):1192-1208; DECEMBER 2000 DOI: 10.1080/01621459.2000.10474320

Sugarman, Stephen D. (1996) *Should Congress Engage in Tort Reform* 1 Mich. L. & Pol'y Rev. 121 (1996), Available at: http://scholarship.law.berkeley.edu/facpubs/408 date accessed: 22 Octover 2012

Taleb, N. (2010). *The black swan : the impact of the highly improbable* (2nd ed.). New York: Random House Trade Paperbacks.

Tassey, G. (2002). The economic impacts of inadequate infrastructure for software testing. NIST. RTI, Health, Social, and Economics Research May 2002]

Telang, R., & Wattal, S. (2004). *Impact of Software vulnerability announcements on the market value of software vendors: An empirical investigation*. Social Sciences Research Network. more information needed Workshop on the Economics of Information

Security, 2005, Cambridge, MA, available online, at http://infosecon.net/workshop/pdf/telang_wattal.pdf

Thompson, P., & McHugh, D. (1995). *Work organisations: A critical introduction* (2nd ed.). London: Macmillan.

Toohey, S. (1995). "Competency-based management education: What does it have to offer? " *Asia Pacific Journal of Human Resources, 33* (2), 118–126.

Turnbull, I. (2004). "Privacy in the Canadian workplace: Best practices". *HR Privacy 2004: Managing the New Challenges, Society for Human Resource Management/ HR Technology*.

Turnbull, Shann, (2004) *How US and UK Auditing Practices Became Muddled to Muddle Corporate Governance Principles* (November 4, 2004, Revised May 12, 2005). Available at SSRN: http://ssrn.com/abstract=608241

Van Hove, L. (2014). "Metcalfe's law: not so wrong after all". *NETNOMICS: Economic Research and Electronic Networking, 15* (1), 1–8. doi: 10.1007/s11066-014-9084-1.

Varian, H. (2004a). "System reliability and free riding". In *Economics of Information Security. Advances in Information Security, 12*, 1–15.

Varian, H. (Ed.). (2004b). *System reliability and free riding* (Vol. 12): Kluwer Academic Publishers.

Varian, H. R. (2000). "Managing online security risks". *The New York Times*. 1 June, 2000 -

http://people.ischool.berkeley.edu/~hal/people/hal/NYTimes/2000-06-01.html

Verner, J. (1977). "On the adaptive significance of territoriality". *American Nature, 111*, 769–775.

Walker, J.W. (1992). Human Resource Strategy. McGraw-Hill, New York.

Weigelt, K., & Camerer, C. (1988). "Reputation and corporate strategy: A review of recent theory and applications". *Strategic Management Journal, 9* 5, 443–454. doi: 10.1002/smj.4250090505.

White, D., & Dolin, S. (2006). *Limiting vulnerability exposure through effective patch management: Threat mitigation through vulnerability remediation.* Unpublished Master of Science thesis, Rhodes University.

Williams, P., Dunlevy, C., & Shimeall, T. "Intelligence analysis for Internet security". Retrieved from http://www.cert.org/archive/html/Analysis10a.html

Williams, R. S. (2002). *Managing employee performance: design and implementation in organisation* (2nd ed.). Thomson Learning. London

Winkler, Ira, (1999) "AUDITS, ASSESSMENTS & TESTS (OH, MY)", Corporate Espionage (Prima, 2nd ed.).

Wood, C. C. (1995). “Background checks for employees in computer-related positions of trust (A further contribution on security system checks for employees) ”. *Information Management & Computer Security, 3* (5), 21–22.

Wood, C. C. (1997). Securely handling staff terminations. *Information Management & Computer Security, 5*(3), pp. 21-22.

Wright, C. S. (2008). *The IT regulatory and standards compliance handbook: How to survive information systems audit and assessments*. Syngress Publishing. Burlington, MA

Wright, C. S. (2010a). “The not so mythical IDS man-month: Or Brooks and the rule of information security”. Paper presented at the International Symposium on Software Reliability Engineering - San Jose, CA, USA 1-4 November 2010. http://www.datagyan.com/issretest/content/fast-abstract-papers#1.

Wright, C. S. (2010b). “Of black swans, platypii and bunyips: The outlier and normal incident in risk management”. *SANS Reading Room*.

Wright, C. S. (2010c). “Software, vendors and reputation: An analysis of the dilemma in creating secure software”. Paper presented at the *Intrust 2010*, Beijing, China.

Wright, C. S. (2011a). “A comparative study of attacks against corporate IIS and Apache web servers”. *SANS Reading Room*.

Wright, C. S. (2011b). “Criminal specialization as a corollary of rational choice”. Paper presented at the International Conference on Business intelligence and Financial Engineering, 12-13 December 2011], HK, China.

Wright, C. S. (2011c). *Current issues and liability facing Internet intermediaries.* Paper presented at the ICBIFE, HK, China.

Wright, C. S. (2011d). “A preamble into aligning Systems engineering and Information security risk”. *SANS Reading Room*.

Wright, C. S. (2011e). “Who pays for a security violation? An assessment into the cost of lax security, negligence and risk, a glance into the looking glass”. Paper presented at the *2011 International Conference on Business Intelligence and Financial Engineering* (ICBIFE 2011), Hong Kong.

Wright C (2012f), “Effective Strategies to Manage People and Processes to Leverage Current Investment in Security”, ACS Journal (Presented in an extended form as the conclusion) - Safe and sound. Information Age September/October 2012 |, Pp. 68–-69.

Wright, C. S. (2012a). Safe and sound. *Information Age September/October 2012* |, 68-69.

Wright, C. S. (2012b). “Territorial behaviour and the economics of botnets”. Paper presented at the *SECAU,* 3-5 December, 2012, Novotel Langley Hotel Perth, WA.

Wright, C. S., & Zia, T. A. (2010). “The economics of developing security embedded software”. Paper presented at the *SecAU 2010* Security Congress, 30 November - 2 December 2010. the Duxton Hotel, Perth, WA. Aust.

Wright, C. S., & Zia, T. A. (2011a). “Compliance or security, what cost? ” (Poster). Paper presented at the *ACISP* 2011, Australasian Conference Information Security and Privacy -11-13 July 2011, Melb. Aust.

Wright, C. S., & Zia, T. A. (2011b). A quantitative analysis into the economics of correcting software bugs. *Computational Intelligence in Security for Information Systems*, 198-205.

Wright, C. S., & Zia, T. A. (2011c). “A quantitative analysis into the economics of correcting software bugs”. Paper presented at the Computational Intelligence in Security for Information Systems, Torremolinos - Malaga 8-10 June 2011, Spain.

Wright, C. S., & Zia, T. A. (2011d). “Rationally opting for the insecure alternative: Negative externalities and the selection of security controls”. *Computational Intelligence in Security for Information Systems*, Volume 6694 Pp 206–213.

Wright, C. S., & Zia, T. A. (2011e). “Rationally opting for the insecure alternative: Negative externalities and the selection of security controls”. Paper presented at the *Computational Intelligence in Security for Information Systems, CISIS 2011*, Spain.

Wright, C. S., & Zia, T. A. (2011f). “Using checklists to make better best”. Paper presented at the *9th Australian Information Security Management Conference (Secau Security Congress 2011)*. Perth, Australia.

Zey, M. (1998). *Rational choice theory and organizational theory: a critique*. Thousand Oaks: Sage Publications.

Zhao, H.; & Tsiatis, A. A.; (1997) "A consistent estimator for the distribution of quality adjusted survival time" Biometrika 84 (2): 339-348 doi:10.1093/biomet/84.2.339

Zhu, H., Zhang, Y., Huo, Q., & Greenwood. (2002). “Application of hazard analysis to software quality modelling”. Paper presented at the *26th Annual International Computer Software and Applications Conference.* 26-29 August, 2002, Oxford, England

Glossary

A

Access control lists (ACLs). Ordered lists of firewall filtering rules that specify which packets should be allowed or denied.

Access control policy. Policy specifying access permissions (authorisation) rules for a resource.

Access controls. After initial identification and authentication, access controls allow users to access files, applications and perform certain tasks. Essentially "access controls" control individual access to computer capabilities. They allow the administrator of a computer to customise and define the rights of individual users of that computer, or computers, on a network. Using access controls, the administrator can define who has access to run which applications, view which files or perform certain tasks.

Access permissions (authorisations). These define whether a role or individual should have any access at all and, if so, exactly what the role or individual should be allowed to do to the resource.

Accountability. Ensuring that if misbehaviour happens, it will be clear who is responsible.

ACLs (Access Control Lists). Ordered lists of firewall filtering rules that specify which packets should be allowed or denied.

Administrator. Superuser account in Windows.

Adversary. An opponent; someone who attacks you.

Air gap. Extreme protection technique in which a network is not connected to other networks. Especially useful in military security.

Alarm. Notification when an attack appears to be occurring. May only be issued for incidents above a certain severity level.

Alert. A formatted message describing a circumstance relevant to network security. Alerts are often derived from critical audit events.

Ankle-Biter / Script Kiddie A person who aspires to be a hacker/cracker but has very limited knowledge or skills related to systems. Usually associated with young teens who collect and use simple malicious programs obtained from the Internet[lxxii].

Annual cost of protection. Annual threat severity minus annual cost of countermeasures.

Annual threat severity. Expected cost of an attack per year.

Antivirus programs. Software designed to prevent viruses from spreading onto user computers and servers by filtering out viruses (and usually worms, Trojan horses, and other attack content).

Apache. Widely used webserver application on UNIX (including LINUX) computers.

Asymmetrical warfare. The company should close all security holes; the attacker need find only one that is not closed.

Attack. An attempt to bypass security controls on a computer. The attack may alter, release, or deny data. Whether an attack will succeed depends on the vulnerability of the computer system and the effectiveness of existing countermeasures.

Attack automation. The use of a program that can carry out attacks without human intervention.

Attack software. Victimisation software that allows an attacker to use a compromised computer to attack other computers.

Attacker-in-the-middle attack. Attack in which the attacker intercepts messages going between two parties. May read, delete, or modify messages after interception.

Audit logs. Log files that record who took what actions and when these actions were taken.

Audit Trail. In computer security systems, a chronological record of system resource usage. This includes user login, file access, various other activities, and whether any actual or attempted security violations occurred, legitimate and unauthorised.

Audit. 1. When an attack team hired by the firm attempts to penetrate the system to identify security weaknesses. 2. When an auditor seeks to find problems in the way an organisation is implementing security.

Authenticate. Prove the identity of someone claiming to be a particular person.

Authentication. Identity verification. Often required to gain access to computer systems or networks. For example, authentication is achieved when a user provides their username and password to log onto their ISP.

Availability. Assuring information and communications services will be ready for use when expected.

B

Back Door. A hole in the security of a computer system deliberately left in place by designers or maintainers. Synonymous with trapdoor; a hidden software or hardware mechanism used to circumvent security controls.

Backup. Periodic archival copying of program and data files to a storage medium.

Baselines. Prescriptions that go into detail about how a specific standard should be implemented with a specific technology.

Best practices. Descriptions of what the best firms in the industry are doing about security.

Black hat hackers. Individuals who break into corporate networks for their own benefit.

Blended threats. Automated attacks that combine the features of viruses, worms, and nonmobile malware; spread in multiple ways.

BOGON, See http://www.team-cymru.org/Services/Bogons/

Breach. The successful defeat of security controls which could result in a penetration of the system. A violation of controls of a particular information system such that information assets or system components are unduly exposed.

Buffer overflow. If the attacker sends a message with more bytes than the programmer had allocated for a buffer, the attacker's information will spill over into other areas of RAM. This is a buffer overflow.

Buffer. Section of RAM programs use to store information temporarily.

Bug. An unwanted and unintended property of a program or piece of hardware, especially one that causes it to malfunction.

Business continuity plan. Specifies how a company plans to restore core business operations when natural or human-made disasters occur. Broader than IT-oriented disaster recovery.

Business process analysis. Identifying, describing, and prioritising a firm's major business processes.

C

CERT, A CERT is a Computer Emergency Response Team and includes groups such as CERT Australia and CERT.org.

CIA. An acronym for confidentiality, integrity, and availability.

COBIT v 4.1 is the computer control objectives and standard maintained by ISACA at http://www.cobitonline.info

Comprehensive security. The effort to close *all* avenues of attack.

Compromise. An intrusion into a computer system where unauthorised disclosure, modification or destruction of sensitive information may have occurred.

Confidentiality. The assurance of safeguarding private communications against unwanted eavesdropping.

Configuration. Most programs and devices have optional settings. Configuration is the choosing of specific optional settings.

Containment. In incident response, preventing the situation from becoming worse.

COSO, Committee of Sponsoring Organisations of the Treadway Commission.

Cost of repair. How much it would cost to restore an asset to its previous secure state.

COTS. Consumer off the shelf software.

Credentials. Things such as user names and passwords used in authentication.

Cyberspace. Describes the world of connected computers and the society that gathers around them. Commonly known as the Internet.

Cyber-terror. Attacks by nongovernmental groups that focus on a country's IT infrastructure and its physical infrastructure; in the latter case, attackers may use computers to assist in the physical attack.[lxxiii]

Cyber-war. Conflict in which a country's military makes a concerted attack upon another country's IT infrastructure, physical infrastructure, or both. A major goal of cyber-war is to inflict a massive amount of damage in a brief amount of time, often in conjunction with a traditional physical military attack.

D

DDoS (Distributed DoS). Denial-of-service attack that hits a victim with streams of messages from many compromised computers.

Defence in depth. Exists when the attacker is restricted to breaking through multiple countermeasures to succeed.

Demilitarized zone (DMZ). An IP subnet that contains hosts and firewalls that must be accessed by external hosts.

Denial of Service. Action(s) that prevent any part of a system from functioning in accordance with its intended purpose.

Denial of Service attack (DoS). Action(s) that prevent any part of a system or network from functioning properly. Denial of service can result when a system, such as a Web server, has been flooded with illegitimate requests, thus making it impossible to respond to real requests or tasks.

Disaster recovery. The technical and procedural aspects of how a company can get IT back into operation using backup facilities. More specific than business continuity planning.

Discounted cash flow analysis. Way of computing current value or internal rate of return of benefits and costs taking place over multiple future years.

Distributed IDS. Intrusion detection system that can collect data from many devices at a central manager console (client PC or UNIX workstation).

DMZ (demilitarized zone). An IP subnet that contains hosts and firewalls that must be accessed by external hosts.

DNS (Domain Name System or Domain Name Server). A domain name look-up system which interprets the domain name of a computer that is connected to the Internet into an IP address. DNS servers or switching stations are located at numerous strategic places to assist in the process of routing of e-mail and Internet connections. Successful routing can require routing and switching through several levels of DNS servers.

DoS. See Denial of Service.

DShield. http://www.dshield.org

Due diligence. Investigating the implications of inter-organisational systems closely before agreeing to them.

E

Egress filtering. Stopping attack packets from going out of a site by filtering them out at a border firewall.

Elevating privileges. Being able to do things that should be possible only if a user has higher access permissions than their account was assigned. An example would be accessing files that are only assigned to one group by assuming the rights of another.

Escalation. The act of declaring an incident or apparent incident to be more severe than previously thought; may trigger certain actions.

Espionage. Penetrating a company to learn information useful to a spy's employer.

Ethical hacking. Hacking per a hacker code of ethics. Still illegal unless authorised by the target.

Event correlation. The analysis of simultaneous and sequential events from many IDSs across the network.

Exploit. Noun: Attacker tool (usually a program) for exploiting a known weakness. Verb: To take advantage of a known vulnerability to attack a system.

Extortion. Threatening to divulge sensitive information or do damage if a company does not pay the extortionist.

F

False acceptance rate (FAR). The percentage of applicants who should be rejected but who are accepted.

False alarm. Apparent security incident that turns out to be innocent activity.

False Negative. Occurs when an actual intrusive action has occurred but the system allows it to pass as non-intrusive behaviour.

False Positive. Occurs when the system classifies an action as anomalous (a possible intrusion) when it is a legitimate action.

False rejection rate (FRR). The percentage of applicants who should be admitted but who are rejected.

FAR (false acceptance rate). The percentage of applicants who should be rejected but who are accepted.

Fault Tolerance. The ability of a system or component to continue normal operation despite the presence of hardware or software faults.

Firewall. A system or combination of systems that enforces borders between two or more networks. A firewall regulates access between networks per a specific security policy. It is almost like an invisible barrier that protects a network or computer. The technology is very like its real world equivalent.

Fix. A way to protect against vulnerability. Includes patches, workarounds, and updates.

FRR (false rejection rate). The percentage of applicants who should be admitted but who are rejected.

G

Grey hat hackers. These individuals engage in both white hat and black hat hacking at different times.

Guidelines. Discretionary prescriptions that must be considered but do not have to be followed if there is a valid reason not to follow them.

H

Hacker. A person who holds a great deal of knowledge and expertise in the field of computing, and who can exercise this expertise with great finesse. This individual explores the details of computers, including security holes, and may exploit them. The hacker term has changed meaning over time. It was previously used to describe a dedicated programmer or devoted programming hobbyist.

Hacking. Unauthorised use, or attempts to circumvent or bypass the security mechanisms of an information system or network.

Host. A single computer or workstation; it can be connected to a network.

Host firewall. Firewall software installed on a client or server host to protect that host.

Host IDS. Intrusion detection system that works on data collected on a host computer. The three types of host IDS are protocol stack monitors, operating system monitors, and application monitors.

I

Identification. 1) In authentication, used to determine the identity of a person by comparing their credentials against all users in an authentication database. See Verification. 2) IP header value used in the reassembly of fragmented IP packets. All fragments from the same original have the same identification field value.

IDS. See intrusion detection system.

IIS. See Internet Information Server.

Incident analysis. After a potential security incident has been reported, determining if it is real and how severe it is.

Incident severity. How damaging a security incident is.

Incident. An event in which security is breached successfully by an attacker.

Information warfare. Another name for cyber-war. Attacks by governments that focus on a country's IT infrastructure and its physical infrastructure; in the latter case, attackers may use computers to assist in the physical attack.

Insurance. Arrangement in which an insurance company charges an annual premium, in return for which it will pay for damages if a threat materialises.

Integrity. Assuring information will not be accidentally or maliciously altered or destroyed.

Intellectual property. Proprietary corporate information that should not be divulged outside the firm. Increasingly used for copyrighted material.

Intelligence. Information about enemy intentions and troop dispositions.

Internet Information Server. Microsoft Web Server Software.

Internet Worm. A worm program (see: Worm) that was unleashed on the Internet in 1988. The worm was written by Robert T. Morris as a purportedly benign experiment that had unintentionally disastrous consequences.

Internet. The global Internetwork.

Intranet. Private Internet within a corporation—uses TCP/IP standards.

Intrusion. Any set of actions that attempts to compromise the integrity, confidentiality or availability of a resource.

Intrusion detection system (IDS). A device that warns administrators if it detects a possible attack underway. Also, collects data on suspicious packets for subsequent analysis. Sometimes acts on its own to stop an attack; software and sometimes hardware that captures network and host activity data in event logs and provides automatic tools to generate alarms, and query and reporting tools to help administrators analyse the data interactively during and after an incident.

Intrusion Prevention System (IPS). An IDS differs from an Intrusion Prevention System (IPS) in that an IDS monitors and alerts on potentially malicious data whereas an IPS actively blocks or filters data.

Internal Rate of Return. Also, called the economic rate of return (ERR), this value is a rate of return used in capital budgeting to measure and compare the profitability of investments.

IP, Internet Protocol

J

Java. Popular programming language for creating small programs, called applets, that can be executed on a webpage.

K

Known vulnerability. Software security weakness that has been widely reported.

L

LAMP, Linux, Apache, MySQL, PhP

LAN (Local Area Network). Customer premises network limited to computers in an office, a building, or a campus.

Likelihood of a threat. The probability that a threat will occur and how often it is likely to occur.

Log entry. Event entry in a log file. Each event has a *time stamp* and an *event type*. Beyond that, log files may have other information to help diagnose the event.

Logging. The recording of essential information about events.

M

Macro virus. A type of computer virus that is encoded as a macro and embedded in a document. Macro viruses are commonly associated with Microsoft Office applications. Once the macro virus infects one document, it can embed itself in all future documents created within the applications. Macros may insert words or numbers into documents or change the command functions of the application.

Malicious code. Hardware, software or firmware that is intentionally introduced to a system for an unauthorised or malicious purpose. A Trojan horse is an example of malicious code.

MitM, Man in the middle.

Mission-critical. Capable of stopping the firm's operations, either temporarily or permanently.

Multi-pronged attacks. The simultaneous implementation of multiple IT attacks, each using a different attack method, to maximise destruction and to confuse defenders.

N

NAP, Network Access Protection, See:
http://en.wikipedia.org/wiki/Network_Access_Protection

NAC, Network Access Control:
http://en.wikipedia.org/wiki/Network_Access_Control

Network Level Firewall. A firewall in which traffic is examined at the network protocol (IP) packet level.

Network. Two or more machines interconnected for communications.

P

Packet Filtering A feature incorporated into routers and bridges to limit the flow of information based on pre-determined communications such as source, destination, or type of service being provided by the network. Packet filters let the administrator limit protocol specific traffic to one network segment, isolate email domains, and perform many other traffic control functions.

Packet Sniffer. A device or program that monitors the data traveling between computers on a network

Packet. Message at the Internet layer.

Patch. Piece of software to fix a vulnerability.

Penetration Testing. The portion of security testing in which the evaluators attempt to circumvent the security features of a system. The evaluators may be assumed to use all system design and implementation documentation that may include listings of system source code, manuals, and circuit diagrams.

Penetration. The successful unauthorised access to an automated system.

Port scanning. Scanning a range of TCP port numbers, UDP port numbers, or both for a single host IP address to identify services running on the host.

Principle of least permissions. States that each user should be given the minimum possible permissions to be able to do their work.

Procedures. Prescriptions that specify the actual steps that must be taken by an employee. Procedures go beyond technology to include the actions that humans must take.

Proxy. A firewall mechanism that replaces the IP address of a host on the internal (protected) network with its own IP address for all traffic passing through it. A software agent that acts on behalf of a user: typical proxies accept a connection from a user, decide as to whether the user or client IP address is permitted to use the proxy, perhaps does additional authentication, and then completes a connection on behalf of the user to a remote destination.

Q

Qualitative threat analysis. Aspects of threat damage that are important but difficult or impossible to quantify.

R

Reliable. A protocol that performs error correction.

Repair. Undoing the damage caused by a successful virus attack or some other type of attack.

Risk acceptance. Implementing no countermeasures and absorbing any damages that result if a threat occurs.

Risk reduction. Taking active countermeasures, such as installing firewalls and hardening hosts.

Risk transference. Having someone else absorb the risk, typically through insurance.

ROI (Return on investment). Method for calculating the value of security investments.

S

SCADA (supervisory control and data acquisition). A SCADA system is a type of industrial control systems (ICS). These are computerised systems designed to monitor and control industrial, infrastructure, or facility-based processes (Radvanovsky, Wright, Brodsky, Weiss, & Harley, 2012).

Scope of an asset. Number of functions affected by an asset.

Script kiddie. Someone who uses an attack script created by someone else and who does not have the skills to hack independently.

Secure Shell (SSH) A completely encrypted shell connection between two machines protected by a super long pass-phrase.

Security. A condition that results from the establishment and maintenance of protective measures that guard against hostile acts or influences.

Security Audit. A search through a computer system for security problems and vulnerabilities.

Security baseline. Specific set of actions for making a program or computer secure.

Security breach. Successful attack.

Security Incident. Any act or circumstance that involves classified information that deviates from the requirements of governing security

publications. For example, compromise, possible compromise, inadvertent disclosure, and deviation.

Security Policies. The set of laws, rules, and practices that regulate how an organisation manages, protects, and distributes sensitive information.

Security through obscurity. The false belief that you are safe if you are not well known or have a poorly documented security system.

Social engineering. Tricking an employee into giving out information or taking an action that reduces security or harms a system.

Spam. The functional equivalent to unsolicited, electronic junk mail. It is often used to advertise products or to broadcast a political or social commentary. Spam floods a user's inbox with irrelevant, unwanted messages.

Spoofing. Faking the sending address or otherwise masquerading as an authorised user to gain illegal entry into a secure system.

Spyware. Victimisation programs that communicate with the attacker, sending back information from the compromised computer, including social security numbers, passwords, and other sensitive information.

SSH. See Secure Shell.

SSL (Secure Sockets Layer). Provides authentication and confidentiality on top of existing applications like Web browsers. Digital certificates and digital signatures utilise this protocol layer to enhance security during online transactions.

T

Target-of-opportunity attacks. Attacks that hit firms randomly, such as most virus attacks.

Threat severity. The estimated cost of an attack—the cost of a successful attack times the probability of a successful attack.

Topology. The map or plan of the network. The physical topology describes how the wires or cables are laid out, and the logical or electrical topology describes how the information flows.

U

UNIX. Family of operating systems used primarily on workstation servers but increasingly on PCs (primarily under the name LINUX).

Unnecessary services. Programs that do not need to be running; may contain vulnerabilities and hence should be turned off.

Untrusted network. Network whose traffic must be inspected carefully; for instance, the Internet.

Update. Install a newer version of a piece of software; often fixes vulnerabilities in older versions.

Upgrade. Install a newer version of a piece of software; often fixes vulnerabilities in older versions.

V

Value of protection. The cost of the threat severity minus the countermeasure cost.

Virus. A program that can "infect" or "contaminate" other programs by modifying them to include a copy of itself. Viral code is typically malicious and detrimental to data or system integrity.

Vulnerability assessment tools. Programs that attempt to find weaknesses in a firm's protection suite, giving the systems administrator an understanding of what work still needs to be done.

Vulnerability. Hardware, firmware, or software flows that leave a system open for potential exploitation. A weakness in automated system security procedures, administrative controls, physical layout, internal controls, and so forth that could be exploited to gain unauthorised access to information or disrupt critical processing.

Vulnerability testing. Testing in which vulnerability assessment tools are turned on the corporate network by authorised testers to find vulnerabilities.

W

White hat hackers. 1. Hackers who break into corporate networks but tell network administrators or the vendor of the security system they compromised how they broke into the network, generally to aid them in preventing such an attack in the future. (Such actions are nevertheless illegal unless the hacked firm has given prior permission.) 2. Hackers who attack only as part of approved auditing efforts.

Worm. An independent program that replicates itself, crawling from machine to machine across network connections. It frequently clogs networks as it spreads (often via e-mail.)

Z

Zombie. In distributed denial-of-service attacks, one of many compromised computers that attacks a victim.

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W

Z

A 1 Appendix

Using standard commercial audit methodologies, all systems were tested with unlimited time and resources against standard practice times. This created the baseline for the data collection process, and thus the control for this experiment. To determine if any errors and discrepancies exist in the data, a detailed control set of all known vulnerabilities on the systems was completed using the S.C.O.R.E. methodologies for secure systems. The highest levels of methodology were used for testing though this far exceeds the normal requirements of a secured server in most operational systems.

A challenge to the control set used would have to be based on the strength of the control used. S.C.O.R.E. is a publicly available peer reviewed system. The Centre for Information Security [CIS] and SANS developed and maintained this methodology for this reason. As the S.C.O.R.E. methodology is generally accepted within the information security community and is considered best practice.

The systems were configured on an existing corporate network to simulate a realistic audit for this experiment. Permission was obtained to use these hosts and this network prior to the initiation of the experiment.

Microsoft hosts were updated using Windows Update and Linux and Solaris Hosts had their respective automated patch tools running until the start of the tests.

For the honeypot tests, the network was designed to simulate a corporate network. A single test was conducted against all non-Microsoft Systems. The tests on the Microsoft Systems were completed three times in the following configurations:

1. Default (Out of the Box) Security Levels

2. Microsoft Secure Server Templates to be applied

3. Microsoft Secure Server – High Security Template to be applied and the host will be configured to S.C.O.R.E. Level 1 (defined as the *Minimum due care security configuration recommendations*)[lxxiv].

The penetration tests using Nessus were run on the same systems a second time, independently to verify the results. The same data was collected in both instances. One interesting result of the experiment involved the relative times to complete the penetration tests against the various Windows systems. It was determined that scans of the more highly secured Windows systems took a greater time to complete and the scans against the least secured systems.

The main reason for this result was the comparative lack of responsiveness from the secured hosts. As the secured host, did not respond to port scans involving closed TCP ports, the scanning engine quickly ran resources whilst waiting for TCP response packets.

1.1. Data Analysis

The results of the penetration test, audit, and control group testing was loaded into the R statistical package and SPSS (for windows) for additional statistical analysis.

Two sample t tests or ANOVA testing was conducted across the different groups to determine if there were statistical differences between each of the test methodologies. Analysis of variance test was completed to determine t or F ratio and thus linear strength of the relationships between the groups. Additional statistical analysis was conducted on the audit and penetration test to determine if the level of false positives produced in the testing methodology was significant at the ά = 0.01 level.

In each of the tests, the null hypothesis that there are no associations between either of the test types and the control would be rejected if the t or F results at ά = 0.01.

Finally, the Tukey-Kramer coefficient and between pairs shall be analysed if the ANOVA has rejected the null at the Alpha equals 1% level to investigate paired relationships between the audit penetration test and control results.

1.1.1.1 Audit

There are two definitive classes of Audit, internal and external (AICPA). An audit consists of the evaluation of an organisation's systems processes and controls and is performed against a set standard or documented process. Audits are designed to provide an independent assessment through testing

and evaluation of a series of representations about the system or process. An audit may also provide a gap analysis of the operating effectiveness of the internal controls.

External audits are commonly conducted (or at least should be) by independent parties with no rights or capability to alter or update the system they are auditing (AICPA). In many cases, the external auditor is precluded from even advising their client. They are limited to reporting any control gaps and leading the client to a source of accepted principles. Due to these restrictions, an indication of the maturity or a system against an external standard (such as COBIT) is often engaged.

Internal audits involve a feedback process where the auditor may not only audit the system but also potentially provide advice in a limited fashion. They differ from the external audit in allowing the auditor to discuss mitigation strategies with the owner of the system that is being audited.

Neither an internal or external auditor can validly become involved in the implementation or design process. They may assess the level to which a design or implementation meets its desired outcomes, but must be careful not to offer advice on how to design or implement a system. Most crucially, an auditor should never be involved with the audit of a system they have designed and/or implemented.

There is a large variety of audit types. Some examples include SAS 70 (part 1 or 2) audits, audits of ISO 9001,17799:2/27001 controls, and audits of HIPPA controls. There are many different types of audits and many standards that an audit may be applied to.

An audit must follow a rigorous program (Winkler, 1999). A vulnerability assessment as it is commonly run is more correctly termed as a controls assessment. A controls assessment may also be known as a security controls review.

1.1.1.2 Inspection and Reviews

An audit differs from an inspection in that an audit makes representations about past results and/or performance. An inspection evaluates results at the current point in time. For an audit to be valid, it must be conducted per accepted principles. In this, the audit team and individual auditors must be certified and qualified for the engagement. Numerous "audits" are provided without certification; these however are in consequence qualified reviews.

1.1.1.3 Penetration tests and Red Teaming

A penetration test is an attempt to bypass controls and gain access to a single system. The goal of the penetration test is to prove that the system may be compromised. A penetration test does not assess the relative control strength nor the system or processes deployed, rather, it is a "red teaming" styled exercise designed to determine if illicit access can be obtained, but with a restricted scope. The issue is that it is infeasible to prove a negative. As such, there is no scientifically valid manner to determine if all vulnerabilities have been found and this point needs to be remembered when deciding on whether to use a penetration test process.

Cohen (1998) notes in respect to red-teaming organisations "*one of the teams I work with routinely asks whether they are allowed to kidnap anyone to get the job done. They usually get turned down, and they are rarely*

allowed to torture anyone they kidnap". Red teaming is based on nearly anything goes.

The greatest strength of the penetration test lies in its being able to market the need to improve internal controls to internal management. This may seem contradictory, but it is based on perception. Being that the Internet is perceived as the greatest threat to an organisation's security, management are often focused on the firewall and Internet gateway to the exclusion of the applicable security concerns and risks. As such, Penetration tests do help in selling the need for an increased focus on information security, but often at the expense of an unfocused application of these efforts.

A penetration test is of limited value in the greater scheme of a systems information security audit programme due to the restricted nature of the test and the lack of inclusion of many key controls. Contrary to popular opinion, penetration testing does not simulate the process used by an attacker. The attacker is not limited in the level of time or funds in the manner that restricts the penetration tester. Whereas a successful penetration test may note vulnerabilities, an unsuccessful penetration test does not prove the security of a system (Dijkstra, 1976).

"Red Teaming" differs from Penetration testing in that it is designed to compromise or penetrate a site at all costs. It is not limited to any attack vector (such as a VPN or Internet) but rather is an attempt to access the systems in any feasible manner (including physical access). A typical red teaming goals would include objectives such as "steal 100,000 for Big Bank without being caught and deliver the report of how to do this to the executive of Big Bank" or "Copy file X which is marked as secret".

Both government and business have used red teaming for many decades in a variety of areas including physical and logical based testing. At its simplest, it is a peer review concept. Another way to look at it is a method of assessing vulnerabilities. In cases where red teaming refers to the provision of adversarial perspectives, and the design of the red team is not hampered in the matter is that ethical attacks are. There is a little correlation between a red team exercise and an ethical attack.

The formation of red teams (or cells) is a situation unlikely to occur in any ethical attack. Further, internal intelligence is unlikely to be gathered as part of an ethical attack. In this instance is more likely that the ethical attack will consist of an attack against the Internet gateway. An engagement to red team is wider in scope, areas including internal subversion and associated control checks cannot be ignored in this type of test.

Penetration testing, if done correctly, can provide some value in its free-form approach if the limitations to scope inherent in this type of test are understood. When correctly implemented, a penetration test adds a level of uncertainty to the testing. The benefit of this uncertainty is that it might uncover potential flaws in the system or controls that had not been considered when designing the control system. To be of value, a penetration test needs to do more than a simple tool based scan of a system.

Penetration Testing needs to do something novel and unexpected.

There is little similarity between a penetration test, vulnerability assessment, risk assessment or audit. The lack of understanding of these differences often impedes the implementation of effective security controls.

1.1.1.4 Ethical Attacks

Ethical Attacks are a subset of penetration testing. They are designed to externally validate a set of controls in a manner that is thought to simulate an attack against the system. It should be noted that ethical attackers are not actually testing system security in the manner of an attacker due to a variety of restraints. It has been demonstrated (Cohen, 1997) that ethical attacks do far less to categorically qualify security risks than many other forms of testing. They do not for instance take note of internal controls. Many potential vulnerabilities cannot be discovered in a penetration test by the nature of the testing. Next, it needs to be remembered that there is an economic cost associated with ethical attack styled penetration testing. The Ethical attacker is constrained by a budget of time and thus money, the real attacker is not.

Blind testing by its very nature will take longer to complete than auditing a site with access and knowledge of all the systems (Dijstra, 1976) if any level of assurance is required. The review undertaken by the ethical attacker is thus hobbled from the start. It is infeasible to state that the contractor will have more knowledge at the end of a review if it is done as an ethical attack with limited knowledge over a system review with full information.

Being a black box test format, the lack of foreknowledge as to the qualification of value associated with any asset negates the possible assessment of a vulnerability status by an ethical attack process (Dodson, 2005). Rather, the process is designed to determine a subset of all possible control failures, which may lead to a system breach or compromise. This subset can never equal the entire control set of possible hazards and vulnerabilities.

This said ethical attacks do have value. They are useful for process testing. If the systems and security team go through the internal processes, they can use the ethical attack process as a means of determining an estimate of the levels of protection using time based security. This is achieved by measuring the detection time and the response time. These times may than be compared at different periods (such as weekends and nights) to determine the level of protection over the system.

Unfortunately, most ethical attacks are not used as an exercise to quantify the level of protection or risk to a system. Rather they are used as a simple de facto vulnerability assessment.

1.1.1.5 Vulnerability Assessment

A vulnerability assessment is an assessment and gap analysis of a site's or a system's control strengths. A vulnerability assessment is a risk-based process. The process involves the identification and classification of the primary vulnerabilities that may result in a system impact. Often, methodologies such as fault tree analysis or CCA (cause consequence analysis) are employed in this process.

A vulnerability assessment is a critical component of any threat risk assessment (Keong, 2004). Following the vulnerability assessment, an impact analysis is conducted to be used in conjunction with a threat report to provide for an estimation of the organisation's risk to selected attack vectors.

There are various processes and procedures used to provide vulnerability assessments and threat/risk determinations. Some standards such as AS/NZS 4360:2006 are commonly mandated by government organisations (such as the NSW State government in Australia).

Vulnerability assessments are part of a complete risk analysis program (Moore, 2001).

Vulnerability assessments involve the cataloguing of assets and capabilities. The lack of internal knowledge provided in the typical ethical attack process precludes this phase. A vulnerability assessment helps to quantify and discern the level of risk to a system (Linde, 1975).

Vulnerabilities, and potentially threats to these resources are determined in this process, which is not limited to external attacks. This process needs to consider not only external attacks and even internal attacks, but a necessarily must also consider physical threats and many other tests outside the reach of the ethical attack or basic penetration test.

1.1.1.6 Black and White Box Testing

Both vulnerability assessments and penetration tests may be conducted as a white box or black box analysis. A black box analysis is instigated with little or no knowledge of the system being tested. A white box analysis is conducted with all details of the system provided to the tester in advance of the testing process (Dijstra, 1976).

1.1.1.7 Tools Based Scanning

The common perception that running an automated scanner such as Nessus or one of its commercial cohorts is a vulnerability or penetration test is false. The belief that these services act as an audit is even further from the truth.

Most of the so-called penetration tests that are provided are no more than a system scan using tools. A penetration test, if correctly designed and implemented will attempt the use of various methodologies to bypass

controls. In some instances, this may involve the creation of new or novel scripts/programs.

The issue is not that many people commonly use the words interchangeably but that so-called professionals fail to differentiate the terms. Of concern is the use of audit and the designation, auditor. This is as these terms are often restricted in legislation as most jurisdictions have statutory requirements surrounding their use and application.

1.1.1.8 Agreed Procedures Review

Information security systems provide many of the functions that construct a control system. Of concern are controls that limit access to accounting and financial records. This includes records held by systems that provide an e-commerce transaction path. In many jurisdictions, it is an offence to sign off an audit report when you are not a certified auditor. Traditionally the path around this has been not to call the process of testing the system an audit, but rather to call it an agreed procedures review. An agreed procedures review or simply a review is an analysis of controls performed against an agreed process.

1.1.1.9 Acceptance testing

Acceptance testing is one of the final occasions to recognise any risk or exposure in a system (Myagmar, 2005). The development and implementation of an approved, inclusive and prescribed plan will support the successful execution of a solution, with the least interruption to critical systems. The process of acceptance testing is to garnish an acceptance of the changes or introduction of a system.

Acceptance testing is more correctly an audit or qualified review of a set of implementation objectives to ensure that the system meets the required levels of performance or security.

1.1.1.10 Data conversion

Testing a Data Conversion is a two-stage process (AICPA). Initially the planning process associated with the data conversion is reviewed to determine the sufficiency of any proposed controls. The subsequent stage occurs after the conversion process. The aims of this process are to present an independent evaluation as to the completeness and accuracy of the data after the conversion.

Any conversion of data into another form or to another system bears an elevated risk of error, omission or other deviations to the completeness and accuracy of that data. Standard input and process controls are frequently not maintained in the data conversion process. To be successful, any project, which includes a data conversion process, requires that the accuracy and completeness of the conversion process be preserved.

1.1.1.11 The Decision Test of the process

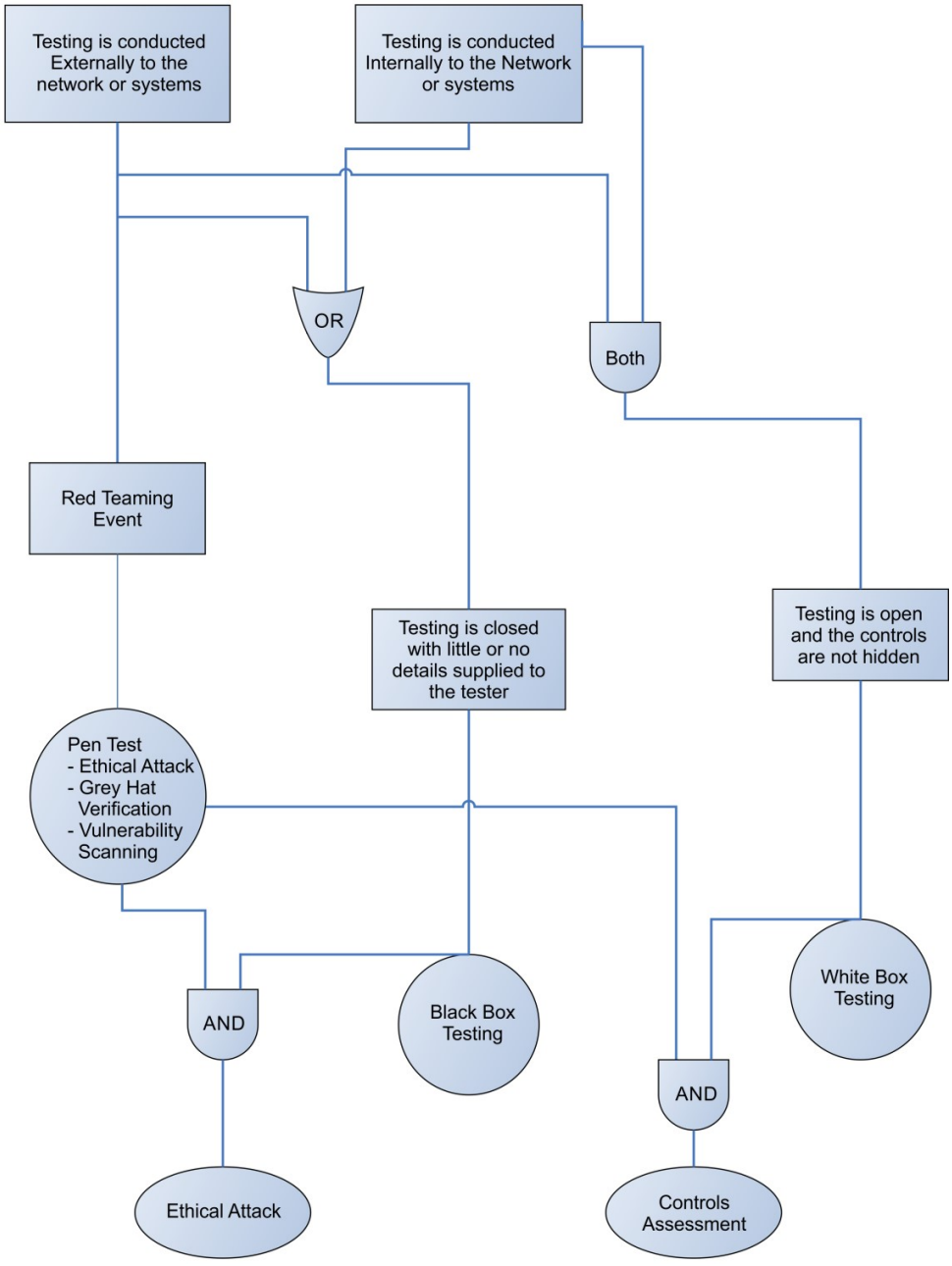

Figure 48. Decision Path of the audit test methodology.

1.1.2 Data Collection

Data was collected from the vulnerability scanners into the following categories:

- Informational
- Low-level vulnerability.
- Medium level vulnerability
- High-level vulnerability

The basis of these methodologies for the levels is developed by SANs, NIST and CIS[lxxv] and is released as S.C.O.R.E. The following table (Table 1 – Vulnerability Levels) provides an explanation on how these levels are determined. Next, the data will be assigned to the following categories:

- Exploitable Externally
- Exploitable Internally
- False Positive

Data is defined to be exploitable externally, where the vulnerable condition may be directly accessed through the Internet by an attacker. Exploitable internally, has been defined as, the condition where the "attacker" must reside inside the firewall to be able to successfully complete the attack on the systems. If the attack is one that may not be concluded successfully from inside the network, it is deemed an externally exploitable attack.

All vulnerabilities, which have been listed in the results, are to be verified manually. Any vulnerability, which has been listed in the report, whether internal or externally exploitable, which in fact cannot be exploited is deemed a false positive.

Table 11 Vulnerability Levels

	Critical	High	Medium	Low	Suspicious
Denial of Service Attack (DOS or DDOS)	Current and continuing loss of service	Possible loss of service if action is not taken	Service could be slightly effected if the attack was to ensue	No loss of service likely to occur	ICMP or large traffic amounts that are unlikely to effect service
Interactive System level compromise	Compromised systems or evidence of such an attempt				
Unauthorised file access/ modification	Compromised systems or evidence of such an attempt	Suspicion of or attempts to access to protected files			
Blocked attacks as noted on the Firewall	Packets that are bypassing the installed firewall policy	Evidence of packet shaping / detailed spoofing in order to bypass firewall rules	Packets targeted at a specific service that may be vulnerable from other sites	General scans	Misc dropped packets
Attacks as noted on the DMZ IDS hosts	System vulnerable to this attack	Targeted attacks on an open service (esp if recently patched)	Detailed probes and continuing scans against specific services	General Scans	
Virus or Worm attacks	Systems infected	Evidence of a virus or worm passing the Anti-virus system	New virus or worm detected	Virus or worm blocked on external anti-virus server	

All high and critical level attacks have been reported in the data table high for analysis purposes. The table field designated as suspicious has been tabulated within the informational data field of the test results. Nessus simplifies the scoring of data. Nessus includes the level of the vulnerability in its scan report output.

It is intended that these results will enable us to summarise the key concerns of this report:

- Total High-level vulnerabilities - Exploitable Externally
- Total System vulnerabilities - Exploitable Externally
- Total High-level vulnerabilities - Exploitable Internally
- Total System vulnerabilities - Exploitable Internally
- Total False Positives

1.2. Methodology – Tools Based External Attacks

1.2.1 Phase 1 – Gain an Understanding of the System

In the first phase of the examination, we:

- Examine your Concept of Operations documentation to gain an understanding of what your system is intended to do, and how it is intended to do it.

- Analyse the network topology and systems configuration documentation, to identify all network devices including servers, workstations, routers and security enforcing devices.

- Examine you Access Requirements (Access Policy) to gain an understanding of what you intend to permit and what you wish to have denied by your system security. This is a very important aspect of the assessment

1.2.1.1 What a Cracker does

To be able to attack a system systematically, a hacker should know as much as possible about the target. Reconnaissance is the first stage. A Hacker will want to get an overview of the network and host systems. Consulting the whois, ripe and arin databases is a good method of gaining information without leaving a trail. Information such as DNS servers used by the domain, administrator contacts and IP ranges routed to the Internet can be obtained. Searching the Usenet for old postings of an administrator may reveal problems, products and occasionally configuration details.

An initial scan of the hosts may show up some interesting services where some in depth researching may lead to interesting attack possibilities. Another issue is looking up possible numbers for the company and trying to connect to a modem. Scanning telephone networks for answering devices and collecting these numbers for a later access attempt may lead to a first entry into the network. Such scans of telephone networks are usually referred to as "war dialling" and were heavily before the Internet existed.

The reconnaissance phase may even consider going through trash bins which is known as “dumpster diving” or visiting loading docks of the target

to collect additional intelligence. During the reconnaissance phase, different kind of tools can be used such as network mapping tools, and vulnerability scanning tools. It is a great help during the attack phase to have an overview about the network.[lxxvi]

Network mapping tools are especially important when doing an internal network assessment as more information is provided than an external scan. For getting a fast report on possible vulnerabilities and security weaknesses, a freeware or commercial vulnerability scanner is useful. These tools scan specified hosts or IP ranges for services and known vulnerabilities. These should be checked as many false positives are often reported.

1.2.2 Phase 2 –Vulnerability Assessment

The vulnerability assessment is conducted to speculate on induced vulnerabilities, which may have been generated by the network's use (or lack) of a certain product, component, or any topology design errors.

Some design and configuration problems we may find within your system are:

- Network topology design not as effective as current industry best practices
- Network management not as effective as current industry best practices
- Configurations not as effective as current industry best practices
- Well-known weaknesses in applications software

- A certain software package or configuration, which has known, exploitable weaknesses, is in use throughout the network;

- Well-known weaknesses in operating systems

- A certain type or family of devices, which has known, exploitable weaknesses, is in use throughout the network;

- Operating Systems configurations not as effective as with current industry best practices

While Phase 2 focuses on identifying weaknesses in the configuration of the networks and systems, an examination of management and administrative approaches is also undertaken.

For example, the vulnerability examination may point out the following types of weaknesses:

- Sensitive data being transmitted across the network in the clear;

- Passwords are not changed on a regular basis;

- Audit trail information is not being collected, or if it is collected, is not being reviewed to identify possible irregularities in system access or usage;

- There are no Security Practices and Procedures document which specifically states the user and administrator security features and responsibilities;

- All weaknesses discovered need be prioritized in readiness for the next Phase.

1.2.3 Phase 3 – Penetration Planning

The penetration-planning phase is where we prepare to conduct the exploits required to compromise the potential vulnerabilities. We identify what vulnerabilities we are going to attempt to exploit and put together a suite of tools in preparation for the next phase, the Penetration Attack.

The tools, which you will use, will consist of:

- Commercially available security tools,
- Publicly available hacker tools

Once you have allocated all the required tools functionality to the penetration plan, you can proceed to Phase 4.

1.2.4 Phase 4 - Penetration Attack

The penetration attack attempts to confirm or discount the presence of actual vulnerabilities from the list of potential vulnerabilities discovered in Phase 2.

In-depth testing will be conducted on the customer's network components, using industry best practice tools and techniques, to identify:

- Confirm the security enforcing functions support any Access Requirements by identifying what is accessible from:
  - Externally, normal public user;

- Internal Restricted Management Segment (if access can be obtained externally);
- Internal Network (if access can be obtained externally)

Using specialist tools attempt to locate an exploit:

- well-known weaknesses in applications software,
- well-known weaknesses in operating systems,
- well-known weaknesses in security enforcing devices,

Additionally, testing will measure the ability of:

- audit capabilities
- system administration practices and procedures
- intrusion detection capabilities
- reaction to intrusions when discovered by audit or intrusion detection mechanisms:
  - incident response plan,
  - contingency plans,

Each confirmed vulnerability should be analysed to:

- Determine the likelihood of someone exploiting the vulnerability, and

- The potential gain by the adversary or loss to your organization.

1.3. Appendix – Threat Risk Assessment Methodology

This is the methodology primarily used for the Vulnerability Assessment / Audit in this research.

In simple terms, a risk is realised when a threat takes advantage of a vulnerability to cause harm to your system. Security policy provides the basis for implementing security controls to reduce vulnerabilities thereby reducing risk. To develop cost effective security policy for protecting Internet connections some level of risk assessment must be performed to determine the required rigour of the policy, which will drive the cost of the security controls deployed to meet the requirements of the security policy. How rigorous this effort must be is a factor of:

- The level of threat an organization faces and the visibility of the organization to the outside world
- The sensitivity of the organization to the consequences of potential security incidents
- Legal and regulatory issues that may dictate formal levels of risk analysis and may mandate security controls for specific systems, applications or data.

Note that this does not address the value of information or the cost of security incidents. In the past, such cost estimation has been required as a part of formal risk analyses to support measurements of the Return on Investment (ROI) of security expenditures. As dependence on public

networks by businesses and government agencies has become more widespread, the intangible costs of security incidents equal or outweigh the measurable costs. Information security management time can be more effectively spent assuring the deployment of “good enough security” rather than attempting to calculate the cost of anything less than perfect security.

For organisations that are subject to regulatory oversight, or that handle life-critical information, more formal methods of risk assessment may be appropriate. The following sections provide a methodology for rapidly developing a risk profile.

It can be prohibitively expensive and probably impossible to safeguard information against all threats. Therefore, modern Information Security practice is based on assessing threats and vulnerabilities and selecting appropriate, cost-effective safeguards. A realistic approach is to manage the risk that these threats pose to information and assets.

It is recognized industry best practice for all organizations to identify their information assets and apply the appropriate security measures based on a Threat and Risk Assessment.

To help organizations meet this requirement, many organizations use an industry standard methodology (like the one below) which has been developed to assess the value of the information that the organization is processing and allows greater flexibility for providing recommended safeguards.

The following diagram illustrates the four-phased approach to performing a Threat and Risk Assessment.

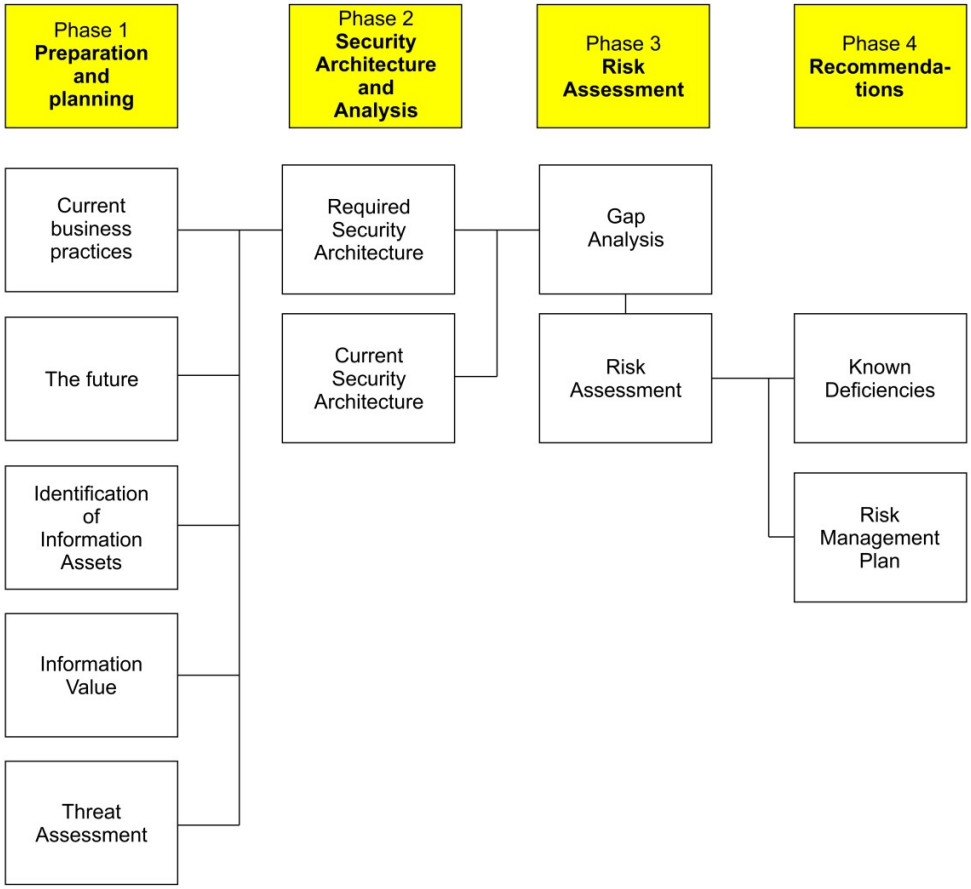

Figure 49 - Risk Assessment Methodology

1.3.1 Phase 1 - Preparation and Identification

1.3.1.1 Current Business Practices

The first step in performing a Threat and Risk Assessment is to define the business practices that are required by the organization to accomplish corporate goals. The Current Business Practices of the organization are documented by analysing the organization's mission statement, corporate plan, type of clients and the services that it provides.

1.3.1.2 The Future

It is critical that the organisation's future business practices and corporate goals are considered throughout the Threat and Risk Assessment process. The plans of the organization must be documented at the start to avoid any possible oversight, preventing the assessment being dated within a short period.

1.3.1.3 Identification of Information Assets

The organization's information assets are identified to determine what should be protected. This requires producing an inventory that lists all information systems and their assets. Each list typically includes the following information:

- the system owner,
- the system's location,
- the nature of business,
- the type of information processed,
- the purpose or application of the system,
- the system configuration,
- the user community, and
- any known inherent strengths or weaknesses of the system.

1.3.1.4 Information Value

After an inventory of the information assets has been produced, a Statement of Sensitivity is documented for each asset. This documents the asset's importance and value to the organization and should reflect its criticality. The statement is produced by analysing the system and the data it processes regarding integrity, confidentially and availability requirements.

1.3.1.5 Threat Assessment

The next step is to identify all threats and threat sources to the organization's information assets and assign a classification that reflects the probability of it occurring. The five levels of threat classification are defined as follows:

- Low: There is no history and the threat is unlikely to occur.
- Low Plus: There is no history and the threat could occur.
- Medium: There is some history and the threat could occur.
- Medium Plus: There is some history and the threat is likely to occur.
- High: There is significant history and the threat is likely to occur.

1.3.2 Phase 2 - Security Architecture Analysis

1.3.2.1 Required Security Architecture

The information gathered in phase I is used to document the business requirements for security within the organization. The key security strategies are identified that will enable the organization to effectively protect its information assets.

Each pre-determined threat to the information assets is matched with an effective safeguard or safeguards. A safeguard is described as many Security Enforcing Functions (SEFs) and associated mechanisms that perform that function are the Security Mechanisms (SM). The process of identifying the required SEFs and the associated mechanisms gives the Organization a security architecture baseline to work towards implementing.

1.3.2.2 Identification of Current Security Architecture

The organization's current security architecture is documented to identify existing Security Enforcing Functions (SEF) and Security Mechanisms (SM). These safeguards and any existing policy or doctrine is identified to produce the current security baseline. This enables identification of differences between the current and required security baselines.

1.3.3 Phase 3 - Risk Assessment

1.3.3.1 Gap Analysis

A gap analysis is performed to highlight any differences between the organization's current security architecture and the required security architecture, determined in phase II of the assessment. The output from this analysis will give the reviewer an indication of the residual risk.

1.3.3.2 Risk Assessment

After the gap analysis has been performed, the determined residual risk should be assessed. This assessment produces a level of risk that is measured by the probability of compromise to the confidentiality, integrity or availability of the designated information system and the data processed on it. Determining the level of risk is completed by comparing the relationship between the threats associated to the residual risks and known vulnerabilities or weaknesses.

1.3.4 Phase 4 - Recommendations

1.3.4.1 Known Deficiencies

Where the assessment of the systems safeguards indicates that they are not able to counter known threats effectively, additional safeguards will be recommended to reduce the risk to an acceptable level. The reviewer will also recommend the type of safeguard required its priority and suggested schedule of implementation.

1.3.4.2 Risk Management Plan

The Threat and Risk Assessment process provides the system manager with an appreciation of the status of the safeguards protecting information assets within his/her organization. An assessment of the adequacy of existing safeguards is performed to provide recommendations to assist the system manager in making an informed decision as to which risks the organization should manage or accept.

The level of acceptable risk is a managerial decision that should be based on the information and recommendations provided in the Threat and Risk Assessment.

1.3.5 Assessment and Conclusion

This methodology has been successful in providing assessments for organizations by producing relevant results. This is achieved by considering the business value of information and the business practices of the organization.

The four-phased approach provides a logical progression, which enables the client to trace through the results from each phase to see how the recommendations were obtained.

1.4. Notes

[i] CMOS worms and BIOS overwriting have been an issue in past malware. CIH (aka Chernobyl) was a virus that overwrote entries in common BIOS chips. In corrupting the BIOS in this manner, the system was left unusable.

[ii] Here we have assumed that a hardware or software compromise has not been built into the system. Cases of infected software disks have been documented, but these are generally accidental distributions by the vendor and can be ignored for the purposes of these calculations.

[iii] http://www.sans.org

[iv] http://www.isaca.org

[v] http://www.owasp.org

[vi] More details of these programs can be obtained at http://en.wikipedia.org/wiki/BonziBuddy and http://www.schneier.com/blog/archives/2011/03/maware_as_job_s.html respectively.

[vii] Figures 1 and 2 are listed as presented at SecAU (Wright & Zia, 2010) and derived from calculations in Tassey (2002).

[viii] A vulnerability market is also known as a marketplace to sell vulnerabilities and exploits or an exploit market.

[ix] Software vendors offer warranties that provide some protection for the user, but they are limited. The vendor cannot account for the actions of the user and a failure to install the software with the use of adequate controls may result in the loss. See Appendix.

[x] Short selling is an investment strategy in which an investor intends to profit from an anticipated *decrease* in each asset price. This involves selling a chattel that the investor does not own through a contract agreement. If the goods sell higher than anticipated, the investor loses money as the goods must be purchased at the (now higher) market rate. In a software risk instrument, a buyer could offset perceived deficiencies in the controls inherent in the software through such a device. A software purchaser could use alternate controls that mitigate the risk assumed to be in a software product and sell the "rights" to have damage from the flaw rectified.

[xi] In each case, it is presumed that the Centre for Internet Security (www.cisecurity.org) standards for installing the software have been reasonably met.

[xii] Which is an extremely low estimate based on the actual pricing of software that has been validated. The creation of a large-scale software product such as Windows without bugs may be economically infeasible at any price.

[xiii] Not only does the market pooled knowledge feed into the price of the derivative, but the vendor's knowledge of their own product does as well.

[xiv] For simplicity, we will assume that this is a vulnerability leading to a catastrophic flaw such as a remote "root" compromise. It is possible to model non-destructive failure processes using "sickness —recovery—death" survival models, but these are outside the scope of this thesis.

[xv] Where the survival time of the first company falls in the expected range of the other company.

[xvi] Through the addition of a "programming system" with the addition of interfaces and a system integration phase.

[xvii] e.g. SANS coding methodology.

[xviii] Which would be quantified through the free market method.

[xix] That is, $Cost(t) = at + b$ where there is an initial cost plus a linear increase in the costs of developing the programmer skills.

[xx] This investment may still be made by the individual programmer with returns to the organisation and for further personal gains (investment in personal capital).

[xxi] Also, reducing the output of the senior staff.

[xxii] F. J. Corbato (MIT): "A long duration project needs to incorporate a turnover rate of at least 20% per annum. New hires need to be technically trained and require time to be able to effectively integrate into the existing team."

[xxiii] See the data and original papers for various derived models.

[xxiv] Ideally, this study would have incorporated international firms, but was restricted due to funding constraints.

[xxv] The term "attacker" has been used in this chapter to refer to the commonly used designations of "hacker" or "cracker" and covers anyone attacking the computer host.

[xxvi] Platypi is the plural for Platypus, an Australian marsupial once believed to be a hoax.

[xxvii] The bunyip is a mythical creature long sought after in Australia. For a long time, it was believed to exist, with the platypus being held as a fraud and myth.

[xxviii] For more detail on these processes see the Springer Series in Reliability Engineering, http://www.springer.com/series/6917

[xxix] See Honeynet Project & Research Alliance.

[xxx] See http://www.sans.org/critical-security-controls/.

[xxxi] See "Vulnerability testing in analogue modem" thread on Security Basics (Securityfocus list).

[xxxii] "Know your enemy – Trend analysis"– The Honeynet Project.

[xxxiii] A scan is defined as a single attempt to gain information on the system from a single host for this experiment. This includes fingerprinting the application as can be done using either a vulnerability scanner or another tool that analyses the service version.

[xxxiv] This is the HTTP server host header. Both IIS and Apache allow an administrator to change or "hide" this system field.

[xxxv] The HoneyNet Project is a research project conducted by the HoneyNet Project & Research Alliance, http://www.honeynet.org .

[xxxvi] Symantec Internet Security Threat Report, January 1 – June 30, 2004.

[xxxvii] "*Survival Time History"* The Internet Storm Centre, The SANS Institute.

[xxxviii] These systems ran the software being tested though they will have no real function.

[xxxix] The test was conducted from February 2010 to December 2010.

[xl] http://cve.mitre.org/data/downloads/allcves.html .

[xli] Definitions used for terms such as an Attack within this document are included in the Appendix below.

[xlii] See http://www.first.org/cvss/, or http://scap.nist.gov/events/2010/itsac/presentations/day1/SCAP_101-CVE_and_CVSS.pdf for a detailed description of this rating system.

[xliii] Phase 1 was set using the default host headers for the web server, Phase 2 involved changing to the alternate host header; "Secure Web Server version 2.3".

[xliv] A higher level of alpha was chosen for the initial test as a lower volume of data had been collected at this point.

[xlv] The two phases of data collection are separated depending on if the server headers have been changed or not. Phase 1 is set using the default host headers for the web server, Phase 2 involved changing to the alternate host header; "Secure Web Server version 2.3".

[xlvi] H_O, There are no differences between the number of attacks against a server type;

H_A, There is a difference between at least one of the tests. Tests conducted at the alpha = 5 level.

[xlvii] A good introduction to the Monte Carlo methods is available at http://www.chem.unl.edu/zeng/joy/mclab/mcintro.html.

[xlviii] The IDS forms a cost function as the increase in reporting results in a greater number of false positives that need to be investigated. In limiting the false positives, the likelihood of missing an incident of note also increases. Each validation of a false positive takes time and requires interaction from an analyst. Hence the tuning of an IDS is balanced on maximising the detection against cost.

[xlix] A mathematical introduction to Bayes' Theorem is available at http://mathworld.wolfram.com/BayesTheorem.html and in more detail from http://plato.stanford.edu/entries/bayes-theorem/ .

[l] See http://www.statsoft.com/textbook/survival-failure-time-analysis/ for an explanation of this form of analysis.

[li] An incident as defined for the purposes of this thesis is an event leading to the failure of the system. This can include a system compromise from an attacker or an infection process of malware (such as a scanning worm).

[lii] A system is defined by an isolated and interactive grouping of computers and processes. This could be a collection of client and server hosts located at a specific location isolated by a common firewall.

[liii] It is assumed that the audit is effective and will uncover an incident if an infected host is reviewed.

[liv] http://www.securecomputing.net.au/News/81027,google-warns-of-web-malware-epidemic.aspx

[lv] Newman NSW.

[lvi] When monitoring the operation of a system or the actions of uses, thresholds are characteristically defined above or below which alerting, alarms, and exceptions are not reported. This range of activity is regarded as baseline or routine activity.

[lvii] Such as currently occurred using Microsoft's MBSA.

[lviii] For instance, a notebook computer could have a set of risks. This would include the risk when connected to the corporate network, when connected to a wireless hotspot etc.

[lix] Such control checks as anti-virus software licenses being up to date and a firewall being installed are common checklist items on most audits. Validating that the anti-virus software is functional or that the firewall policy is effective is rarely conducted.

[lx] Although these services do remain highly elastic for many smaller organisations who may choose not to control risk when budgets are tight.

[lxi] This includes SOX, APRA, FISMA, and many other compliance regimes.

[lxii] Proximity, a notion first established in Caparo Industries Plc. v. Dickman, [1990] 2 A.C. 605, is the initial phase of the assessment. The subsequent phase enquires as to whether there are policy considerations which would reduce or counteract the duty created under the initial stage. Mutually, the phases are to be met regarding the facts of cases previously determined.

[lxiii] PCI-DSS (version 1.1) is the Payment Card Industry Data Security Standard and is contractually required to be adhered to by all merchants that process VISA, MasterCard and other payment card products. This requirement and standard is maintained by the PCI Standards Council at https://www.pcisecuritystandards.org/ .

[lxiv] CIS benchmark and scoring tools are available from http://www.cisecurity.org/ .

[lxv] At the $\alpha = 95\%$ level.

[lxvi] This figure is based on internal data from four years of audit firm data and six years of security company data.

[lxvii] The bug data was collated from Secuna (http://secunia.com/) and is derived using the following products: Microsoft Windows Vista, Windows XP (Home Edition), Windows XP Professional, Windows 2000 Advanced Server, Windows 2000 Datacenter Server, Windows 2000 Professional, Windows 2000 Server, Windows 7, Windows 95, Windows 98, Windows 98 Second Edition, Windows CE .NET 4.x, Windows CE 3.x, Windows CE 5.0 & CE 6.0, Windows Millennium, Windows Mobile 2003, Windows Mobile 5.x, Windows Mobile 6.x, Windows NT 4.0 Server, Terminal Server Edition, Windows NT 4.0 Workstation, Windows Server 2003 Datacenter Edition, Windows Server 2003 Enterprise Edition, Windows Server 2003 Standard Edition, Windows Server 2003 Web Edition, Windows Server 2008, Windows Storage Server 2003.

[lxviii] http://cisecurity.org/en-us/?route=downloads.audittools .

[lxix] SANS 20 Critical Security Controls, Twenty Critical Security Controls for Effective Cyber Defense: Consensus Audit Guidelines (http://www.sans.org/critical-security-controls/)

[lxx] The process involved obtaining official consent. All participants likewise provided signed, explicit written consent for their involvement in this study.

[lxxi] It is likely that the composition of personality types in IT would also vary based on "generational divisions". Little quantitative data on this subject was found to be available in this study. Research into the personality compositions of early entrants, "Generation X" and "Gen Y" IT workers would be warranted.

[lxxii] http://www.techrepublic.com/article/you-say-cracker-i-say-hacker-a-hacking-lexicon/ .

[lxxiii] State-based forms of attack come from funded parties.

[lxxiv] S.C.O.R.E. - http://www.sans.org/score/

[lxxv] CIS, Http://www.cis.org, the Centre for Information Security.

[lxxvi] For this test, all information will be considered available and thus make the "penetration test" phase easier in comparison.