Cape Peninsula
University of Technology

A FRAMEWORK FOR SOFTWARE PATCH MANAGEMENT IN A MULTI-VENDOR ENVIRONMENT

by

GRANT DOUGLAS HUGHES

Thesis submitted in fulfilment of the requirements for the degree

Master of Technology: Information Technology

in the Faculty of Informatics and Design

at the Cape Peninsula University of Technology

Supervisor: Mr Jay Barnes
Co-supervisor: Dr Andre de la Harpe

Cape Town
November 2016

DECLARATION

I, Grant Douglas Hughes, declare that the contents of this thesis represent my own unaided work, and that the thesis has not previously been submitted for academic examination towards any qualification. Furthermore, it represents my own opinions and not necessarily those of the Cape Peninsula University of Technology.

	November 2016
Signed	**Date**

ABSTRACT

Software often requires patches to be installed post-implementation for a variety of reasons. Organisations and individuals, however, do not always promptly install these patches as and when they are released. This study investigated the reasons for the delay or hesitation, identified the challenges, and proposed a model that could assist organisations in overcoming the identified challenges. The research investigated the extent to which the integration of software patch management and enterprise data security is an important management responsibility, by reviewing relevant documents and interviewing key role players currently involved in the patch management process. The current challenges and complexities involved in patch management at an enterprise level could place organisations at risk by compromising their enterprise-data security.

This research primarily sought to identify the challenges causing the management of software patches to be complex, and further attempted to establish how organisations currently implement patch management. The aim of the study was to explore the complexities of software patch management in order to enhance enterprise data security within organisations.

A single case study was used, and data were obtained from primary sources and literature. The study considered both technological and human factors, and found that both factors play an equally important role with regard to the successful implementation of a patch management program within an organisation.

Key words: Hot fix, software patch, Information security, patch management, Information security policy, patch management policy, software vulnerability, zero day vulnerability, data security, information security risk.

ACKNOWLEDGEMENTS

I wish to thank:

- Mr Jay Barnes, academic supervisor
- Dr Andre de la Harpe, academic co-supervisor
- Cape Peninsula University of Technology

TABLE OF CONTENTS

DECLARATION .... **ii**
ABSTRACT .... **iii**
ACKNOWLEDGEMENTS .... **iv**
LIST OF FIGURES .... **x**
LIST OF TABLES .... **xi**
ABBREVIATIONS .... **xii**
GLOSSARY .... **xiv**

CHAPTER ONE: BACKGROUND AND INTRODUCTION **1**
1.1 Introduction 1
1.2 Background 1
1.3 Research problem statement 3
1.4 Research questions, methodology, aims and objectives 4
1.4.1 Research aims 5
1.4.2 Research objectives 5
1.5 Research methodology 5
1.5.1 Research philosophy 5
1.5.1.1 Ontology 5
1.5.1.2 Epistemology 6
1.5.1.3 Research approach 6
1.5.2 Research strategy 6
1.5.2.1 Case study 6
1.5.2.2 Unit of analysis 7
1.5.2.3 Unit of observation 7
1.5.3 Data collection 7
1.5.3.1 Sampling 7
1.5.4 Data analysis 8
1.5.4.1 Transcribing 8
1.5.4.2 Coding 8
1.5.4.3 Themes 8
1.5.4.4 Categories 9
1.6 Delineation 9
1.7 Ethical considerations 9
1.8 Chapter summary 10

CHAPTER TWO: LITERATURE REVIEW ... 12
2.1 Introduction ... 12
2.2 Context of the literature review ... 12
2.3 Confidentiality, integrity and availability of data ... 13
2.4 Threats, risk and vulnerability (TRV) ... 14
2.4.1 Threats ... 14
2.4.1.1 Virus ... 14
2.4.1.2 Eavesdropping ... 14
2.4.1.3 Worm ... 15
2.4.1.4 Backdoor ... 15
2.4.1.5 Direct attack ... 15
2.4.1.6 Denial of Service (DOS) attack ... 15
2.4.2 Risks ... 16
2.4.3 Vulnerabilities ... 16
2.5 Zero-day vulnerabilities ... 17
2.6 The complexities of software ... 19
2.6.1 Software evolution ... 20
2.6.2 Legacy systems ... 20
2.7 Software management methods ... 21
2.7.1 Configuration management methods ... 21
2.7.2 Deployment methods ... 21
2.8 Software patches ... 22
2.9 Patch management ... 22
2.10 Software patch management and related risks ... 23
2.10.1 The importance of patch management ... 24
2.10.2 Patch automation ... 24
2.10.3 Hot patching ... 26
2.10.4 Horisontal patching ... 26
2.10.5 Patch management configuration ... 27
2.10.6 Software patch management challenges ... 27
2.10.7 Release management and patch management ... 29
2.11 The role of software vendors in the patch management process ... 29
2.11.1 Vendors’ behaviour ... 30
2.11.2 Disclosure ... 31
2.11.3 Initiatives and incentives ... 32
2.12 Considerations for developing a patch management strategy ... 33
2.12.1 Sources of patch information ... 33
2.12.2 Roles and responsibilities ... 34

2.12.3 Vulnerability management 34
2.12.3.1 Patch classification systems 34
2.12.4 Testing 36
2.13 Chapter summary 37

CHAPTER THREE: RESEARCH DESIGN AND METHODOLOGY 39
3.1 Introduction 39
3.2 Research philosophy 40
3.2.1 Ontology 40
3.2.2 Epistemology 41
3.2.2.1 Interpretivist 41
3.2.2.2 Positivist 41
3.3 Research approach 41
3.3.1 Inductive 44
3.3.2 Deductive 45
3.4 Research strategy 45
3.4.1 Case study 45
3.5 Data collection 48
3.5.1 Sampling 49
3.5.2 Unit of analysis 50
3.5.3 Unit of observation 50
3.5.4 Interviews 50
3.5.4.1 Semi-structured interviews 51
3.5.4.2 Interview guide 52
3.5.5 Summary of data collection 52
3.6 Data analysis 52
3.6.1 Transcribing 54
3.6.2 Coding 55
3.6.3 Thematic analysis 55
3.6.4 Categories 55
3.7 Ethics 56
3.8 Chapter summary 57

CHAPTER FOUR: DATA COLLECTION AND PRELIMINARY DATA ANALYSIS 59
4.1 Introduction 59
4.2 Data collection 60
4.3 Document review 60
4.4 Description of case 61

4.5 Research sample ... 63
4.6 Sampling ... 64
4.7 Interview process ... 64
4.8 Transcribing ... 64
4.9 Coding ... 64
4.10 Themes ... 65
4.11 Categories ... 65
4.12 Themes emerging from the literature ... 66
4.13 Key findings from interview data ... 67
4.14 Final data analysis ... 68
4.15 Mapping the findings to themes ... 70
4.16 Chapter summary ... 72

CHAPTER FIVE: DISCUSSION ... 73
5.1 Introduction ... 73
5.2 Themes developed ... 74
5.2.1 Sources of software patches and related information ... 74
5.2.2 Patch management policy development stakeholders ... 74
5.2.3 Roles and responsibilities ... 76
5.2.4 The importance and challenges of testing ... 77
5.2.5 Patch deployment prioritising mechanism ... 79
5.2.6 Interviewee familiarity: patch management issues ... 80
5.2.7 Software vendor behaviour and their effect on patch management ... 80
5.2.8 Patch management challenges ... 81
5.2.8.1 Resources, infrastructure and tools ... 82
5.2.8.2 No all-inclusive off-the-shelf solution ... 83
5.2.8.3 Communication ... 83
5.2.8.4 Maintenance timeslots ... 84
5.2.8.5 Legacy systems ... 85
5.2.8.6 The people aspect ... 85
5.2.9 Risks related to patch management ... 86
5.2.10 Consistency of patch management activities ... 87
5.2.11 The importance of patch management ... 88
5.3 Chapter summary ... 89

CHAPTER SIX: RESEARCH QUESTIONS, RECOMMENDATIONS AND CONCLUSION ... 90
6.1 Introduction ... 90
6.2 Patch management challenges ... 91

6.2.1 Resources, infrastructure and tools ... 91
6.2.2 No all-inclusive off-the-shelf solution ... 91
6.2.3 Communication and documentation ... 91
6.2.4 Maintenance timeslots ... 92
6.2.5 Testing ... 92
6.2.6 Legacy systems ... 92
6.2.7 The people aspect ... 93
6.3 Answering the research questions and research sub-questions ... 93
6.3.1 Research sub-questions ... 93
6.3.1.1 Patch management challenges (RSQ 1.1) ... 93
6.3.1.2 Patch management challenges created in a multi-vendor software environment (RSQ 1.2) ... 94
6.3.1.3 Controls to assist with patch management (RSQ 1.3) ... 94
6.3.1.4 Patch management and prioritising (RSQ 2.1) ... 95
6.3.1.5 Identifying relevant software patches (RSQ 2.2) ... 96
6.3.1.6 The role players in the patch management process (RSQ 2.3) ... 96
6.3.2 Research questions ... 97
6.3.2.1 The complexity with patch management (RQ1) ... 97
6.3.2.2 Current patch management implementation within the company (RQ2) ... 99
6.4 Proposed solution/model to the research problem ... 102
6.5 Recommendations ... 104
6.6 Conclusion ... 105
6.7 Future research ... 107

REFERENCES ... 109

APPENDIX A: THREE INTERVIEW TRANSCRIPTS ... 120
APPENDIX B: EXAMPLE OF CODING TECHNIQUE APPLIED (LATENT CODING) ... 153
APPENDIX C: UNSTRUCTURED IDENTIFIED THEMES ... 154
APPENDIX D: STRUCTURED IDENTIFIED THEMES ... 159
APPENDIX E: REPRESENTATION OF RESEARCH SAMPLE ... 164
APPENDIX F: RESEARCH INSTRUMENT: INTERVIEW QUESTIONS ... 165
APPENDIX G: CONSENT FORM: PARTICIPATION IN CASE STUDY RESEARCH ... 167
APPENDIX H: NON-DISCLOSURE AGREEMENT: TRANSCRIPTION ANALYST ... 168
APPENDIX I: INTERVIEWEE QUOTES TO INTERVIEWEE AND TRANSCRIPT MAPPING ... 169

LIST OF FIGURES

Figure 3.1: Four paradigms for the analysis of social theory 43
Figure 4.1: Table of contents: the company’s patch management framework 61
Figure 4.2: Table of contents: the company’s patch management Policy 61
Figure 4.3: High-level layout of the case organisation 62
Figure 4.4: Example of the coding process used... 65
Figure 4.5: Summary of data analysis process.. 67
Figure 6.1: The current patch management model in the company................................... 101
Figure 6.2: The proposed model (The Single Decision Making Model) 103

LIST OF TABLES

Table 1.1: Research problem and research question summary 4
Table 4.1: Research sub-questions and interview guide mapping....................................... 59
Table 4.2: Summary of current and previous job roles of participants 63
Table 4.3: Emerging themes from the literature with supporting authors 66
Table 4.4: Explicit findings (summarised)... 68
Table 4.5: Implied findings (summarised)... 69
Table 4.6: Exceptions to the implicit and implied findings (summarised) 70
Table 4.7: Mapping findings to corresponding themes 70
Table 6.1: Research question and sub-questions summary.. 90

ABBREVIATIONS

Abbreviation	Definition
CIA	Confidentiality Integrity Availability
CIO	Chief Information Officer
COSO-ERM	Committee Of Sponsoring Organisations - Enterprise Risk Management
CPUT	Cape Peninsula University of Technology
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
CWSS	Common Weakness Scoring System
DOS	Denial of Service
ERM	Enterprise Risk Management
FRC	Further Research Community
HP	Hewlett Packard's
HTTP	Hypertext Transfer Protocol
IDS	Intrusion Detection System
IE	Internet Explorer
IEEE	Institute of Electrical and Electronics Engineers
IT	Information Technology
ITIL	Information Technology Infrastructure Library
MDM	Mobile Device Management
NIST	National Institute of Standards and Technology
RACI	Responsible Accountable Consulted Informed
RDF	Resource Description Framework
RSQ	Research Sub-Question
RQ	Research Question
SANS	SysAdmin, Audit, Network and Security

Abbreviation	Definition
SCCM	System Centre Configuration Manager
SQL	Structure Query Language
STRIDE	Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege
TNIPP	The National Infrastructure Protection Plan
TRV	Threats, Risks and Vulnerabilities
URI	Uniform Resource Identifier

GLOSSARY

Term	Description
Information security policy	Höne and Eloff (2002:2) define an information security policy as “a direction-giving document for information security within an organisation”.
Patch management	Patch management is defined as the process of “identifying, acquiring, installing, and verifying patches for products and systems” (Souppaya & Scarfone, 2013:2).
Software patch	A patch is defined as a piece of software that “corrects security and functionality problems in software and firmware” (Souppaya & Scarfone, 2013:2).
Software vulnerability	A vulnerability is defined as “a fault on system requirements or programs which allows an attacker to violate the system integrity” (Okamura, Tokuzane & Dohi, 2009:120).
Zero-day vulnerability	McQueen, Boyer, McQueen and McBride (2009:1) define zero-day vulnerability as “any vulnerability, in deployed software, which has been discovered by at least one person but has not yet been publicly announced or patched”.

CHAPTER ONE: BACKGROUND AND INTRODUCTION

1.1 Introduction

Often, after software has been developed, it does not do what was initially required. New requirements emerge that were not initially planned for, or the software has a security vulnerability that can result in a security challenge, breach and/or risk. A patch is defined as a piece of software that "corrects security and functionality problems in software and firmware" (Souppaya & Scarfone, 2013:2). The implementation of an integrated software patch management system could hold various benefits for an organisation, including but not limited to the addition of new functionality, ensuring on-going vendor support for applications and securing vulnerable software systems. Souppaya and Scarfone (2013:2) state that "patch management is required by various security compliance frameworks, mandates, and other policies".

There are different variants of software patches, as well as methods to deploy these patches. The basic patching principles should be maintained at all times. The software patch process is conceptually simple, yet practically challenging for many organisations (Lahtela & Jäntti, 2011).

Mohr and Rahman (2011) report that Sony was hacked in 2011 due to some of their internet-facing servers running outdated software. Leavitt (2011) confirms that DigiNotar was also hacked due to unpatched software. These examples beg the question: *If deploying patches is as easy as installing a small piece of software, why do big corporate organisations such as Sony and DigiNotar get it wrong to the extent where the software system security is breached?* This research aims to explore the complexities of software patch management in order to enhance enterprise data security within organisations.

The research presents a case study on a Cape Town-based retailer that is representative of the research problem. The case organisation is referred to as 'the company' for the purpose of this discussion. A model was developed to help organisations implement a more efficient patch management process and simultaneously improve data security.

1.2 Background

It often seems as though organisations are either not paying much attention to patch management, or not doing it right (Gerace & Cavusoglu, 2009). Ioannidis, Pym and

Williams (2012) suggest there may be a link between the level of applied patches and the level of data security in an organisation. This statement becomes evident when examining a few examples where unpatched software led to a security breach.

In 2001, Code Red caused havoc to many networks although a patch was released months prior to the attack being launched. According to Moore and Shannon (2002), the worm-spread propagated through the internet by using Hypertext Transfer Protocol (HTTP) requests and exploited a buffer overflow vulnerability that was known at the time. In the instance of Code Red, the amount of time an administrator would have had to test these patches was considerably longer than usual, yet more than 115 000 websites were attacked by Code Red (Moore & Shannon, 2002). The time between vulnerability discovery and patching is often months, or in some instances years (Cavusoglu, Cavusoglu & Zhang, 2008).

The Structure Query Language (SQL) Slammer worm in 2003 highlighted the importance of patching vulnerabilities as soon as possible, because a patch was released six months prior to the worm causing havoc on the internet (Zhou, Leckie & Karunasekera, 2010). According to Bilge and Dumitras (2012) the amount and frequency of released patches could be a potential challenge, as it may make it difficult for organisations to process released patches in a timely fashion. Consequently, critical patches could be overlooked, delayed or exploited before it has been patched.

In 2009 the Sheffield hospital was affected by the Conficker worm. The virus seized the hospital's entire network, leaving it digitally crippled for several days. Although the worm broke out in November 2008, Microsoft released a patch for this specific worm in October 2008 (Microsoft, 2008). This scenario is consistent with the opinion of Zhao, Furnell and Al-Ayed (2009) that not all users and administrators realise and understand the importance of software patch management.

There appears to be a tendency for organisations to not immediately action newly-released software patches. It is possible that organisations are facing challenges that prevent them from maintaining efficient patch management solutions and policies, or lack the understanding of its importance. One of the reasons for not immediately installing a software patch, according to Souppaya and Scarfone (2013:3), was "the significant risk of installing patches without first testing them".

Assuming administrators continue to manage patches in an *ad hoc* manner—as is currently the case—systems may remain vulnerable to exploitation for an extended period. Untested patches may be deployed and cause instability in environments (Raja & Tretter, 2011). Vulnerable systems may be exploited and data integrity and confidentiality compromised (Ackling, Alexander & Grunert, 2011). Gerace and Cavusoglu (2009) indicate that although patches are the solution, the patch management process may sometimes be flawed.

This research aims to explore the complexities of software patch management in order to enhance enterprise data security within organisations. The research further establishes where the responsibility of patching should be placed in an organisation, both in terms of the non-technical and technical activities. As a result, a model is proposed to address some of the challenges with the current patch management process faced within the company. The proposed model could assist with an increased level of data security within the organisation.

1.3 Research problem statement

Software vendors often release patches in large numbers and at irregular intervals. This poses a challenge for organisations to stay abreast of all patches released, decide on which patches to test, those patches to install and those not to install. This could lead to a scenario where critical security patches may not be deployed in a timely manner, leaving specific systems vulnerable to attacks (Bilge & Dumitras, 2012).

Ahmad, Maynard and Park (2014:5) explain that “updating and patching application systems has become a critical preventive technique aimed at denying attackers pathways into the organisation”. The authors clarify that since the number of vulnerabilities is so high and resources are often limited, only a few vulnerabilities can be addressed at any given time. The challenge now resides with the information security manager to decide (i) what should be addressed and according to what priority, and (ii) the level of priority to assign to the vulnerability, keeping in mind the available resources. There are various known patch management challenges that, if not addressed, could lead to security compromises that can be prevented (Souppaya & Scarfone, 2013). Zero-day attacks and limited testing time are two known patch management challenges. Another example is the 2015 Android feat where mobile devices were exploited when receiving a picture without consent or intervention from the device owner. Although Google released a patch for this vulnerability, the company admitted that millions of Android devices had not yet been updated (three months after the release of the patch and at the time of this article) (BBC News, 2015).

Organisations often find themselves running IT systems that may either be unstable or prone to intrusion because of challenges and complexities involved in patch management at an enterprise level.

1.4 Research questions, methodology, aims and objectives

Table 1.1: Research problem and research question summary

Research Problem	Organisations often find themselves running IT systems that may either be unstable or prone to intrusion because of challenges and complexities involved in patch management at an enterprise level.	
Research Question 1 (RQ1)	What causes the management of software patches to be complex?	
Research Sub-Questions (RSQs)	**Research Methods(s)**	**Objective**
1.1 What challenges do organisations face with regard to patch management?	Case Study	To identify the challenges patch management could present for an organisation.
1.2 What challenges are created by a multi-vendor environment?	Case Study	To identify challenges that originates within multi-vendor environments.
1.3 What controls can organisations implement in order to deploy the relevant patches?	Case Study	To identify the effects a lack of proper management could have on data security within an organisation.
Research Question 2 (RQ2)	How do organisations implement patch management in order to enhance enterprise data security?	
Research Sub-Questions (RSQs)	**Research Methods(s)**	**Objective**
2.1 How do organisations prioritise patches for deployment?	Case Study	To determine how organisations prioritise patches offered to them by the industry.
2.2 How do organisations identify patches best suited for their security needs?	Case Study	To identify the benefits that can be obtained when patch management is integrated with information security.
2.3 Who are the key role players in managing the patch deployment process?	Case Study	To identify the stakeholders involved in the patch management process.

1.4.1 Research aims

The aim of the study is to explore the complexities of software patch management in order to enhance enterprise data security within organisations. The objective of the study is to identify some of the challenges organisations face when implementing patch management, to determine how patch management affects enterprise data security, and to determine the challenges faced by organisations with multi-vendor systems.

1.4.2 Research objectives

The objectives of the study are to:

- Identify some of the challenges organisations face when implementing patch management
- Determine the challenges faced by organisations with multi-vendor systems
- Determine how patch management affects enterprise data security

1.5 Research methodology

For the purpose of this study, a single case study was conducted. The selected research methodology required that data be gathered from both primary sources and secondary sources. The primary source for the study was interviews with participants who currently play a role in the patch management process. A literature review was the basis of the secondary source, and a brief overview is presented below.

1.5.1 Research philosophy

1.5.1.1 Ontology

Neuman (2011) describes ontology as the explicit specifications of a shared conceptualisation. It thus contains the type of object or concept that exists, as well as its containing properties and relations.

This research assumed that there is no single truth or understanding of how organisations currently manage their patch deployment process. This research was based on the relativist tradition, which Neuman (2011:107) describes as a principle in “interpretive social science that no single point of view or value position is better than others”. Real-life conditions were explained in the context in which they exist, and in the process, the objective knowledge as was produced.

1.5.1.2 Epistemology

According to Neuman (2011), epistemology aims to answer:

i. What is knowledge?
ii. How is knowledge acquired?
iii. What are the most valid ways to reach the truth?
iv. How do we know what we know?

This study was conducted based on the premise that a world exists independently of the researcher's thoughts and perceptions and as such, drives the research philosophy. In the case of this research, the researcher provided an interpretation of what he found participants doing, and what he believes to be their reasons. The problem was viewed from the perspective of the participants involved and every effort was made to maintain an awareness of the research context.

1.5.1.3 Research approach

According to Dubois and Gadde (2002), a deductive approach is geared at testing a theory, whereas an inductive approach is tailored toward the generation of a new theory that emerges from the data. The authors further state that the inductive approach is normally associated with qualitative research, while the deductive approach is generally associated with quantitative research.

The research methodology requires that data be gathered from literature as well as primary sources. An inductive approach to the research was followed. There was no predetermined hypothesis; however, the research problem informed the research process for the investigation. The factors under investigation were not purposefully manipulated in any manner.

Specific events were described in rich detail. Given the nature of this research, a qualitative approach was adopted. Neuman (2011) describes qualitative research as an approach concerned with recording, analysing and attempting to uncover a deeper meaning in human behaviour and experience. The research followed an inductive approach.

1.5.2 Research strategy

1.5.2.1 Case study

Yin (2013) defines a case study as an empirical enquiry that explores a contemporary phenomenon within its natural context, when the boundaries between the context and the phenomenon are not clearly defined. As a result, multiple sources of evidence are needed. Conversely, Flyvbjerg (2006:2) describes a case study as a “detailed examination of a single example”. Both definitions are consistent with the research method used in this study. However, for the purpose of this research, the definition provided by Yin (2013) was used. This definition best describes the context in which this research was conducted.

The case selected for this case study is a Cape Town-based retail organisation that is representative of the research problem. A single case study was conducted.

1.5.2.2 Unit of analysis

The unit of analysis for this study was the patch process within the organisation. The patch process was researched by means of conducting interviews and reviewing company documents such as the information security policy and the patch management policy.

1.5.2.3 Unit of observation

The unit of observation was the people within the organisation who play a role in the current patch management process. Within the company, three relevant departments were identified: The Head Office environment, the Server environment and the Store environment. The people within these respective environments were the units of observation.

1.5.3 Data collection

1.5.3.1 Sampling

The case study organisation is located in Cape Town. Convenience sampling was used to select the organisation. Neuman (2011) describes convenience sampling as having availability and convenience as the main criteria for selection. The selection of interviewees was informed by the literature, and included personnel from the server team, internal audit, system administrators, security personnel and enterprise architecture. Purposive sampling was used to select interviewees. Purposive sampling is described as “the most common sampling technique”, where “the researcher actively selects the most productive sample to answer the research question” (Marshall, 1996:523). Snowball sampling was also used to identify participants who were not explicitly identified in the literature, and yet were connected to the patch management process either directly or indirectly in this particular case. Participants may not know each other, but information obtained from one may lead to the next.

Semi-structured questionnaires using an interview guide was used to obtain data from key role players in the patch management process within the case organisation. Company documents such as the patch management policy and framework were reviewed prior to the interviews. Interviews were used as the method of data collection. With the necessary consent from the participants, the interviews were recorded. The recorded interviews were encrypted with Microsoft Bit Locker and stored on Google Drive. Only the researcher, academic supervisor, academic co-supervisor and transcription analyst have access to the interview recordings.

1.5.4 Data analysis

1.5.4.1 Transcribing

According to King (1994), many research studies collect data in audio or video, and usually transcribe it into written text for closer analysis. King (1994) explains that transcription is an interpretive and repetitive process. The researcher often had to listen to the recording several times, paying special attention to the tone of voice, hesitations and other distinct observations. In this study, this function was outsourced. However, the researcher verified all recordings against the transcripts to ensure accuracy and to become immersed with the data. For examples of three transcribed interviews, see Appendix A. All other interview transcripts are available on request.

1.5.4.2 Coding

According to Neuman (2011), there are two major types of coding: manifest and latent coding. Manifest coding is described as a technique where the author may count the number of times a word or phrase is mentioned. This process can be achieved with the help of computer software. Latent coding is where the researcher would look for the underlying meaning in the text. In this instance an entire paragraph would be read, and a possible theme could be identified. In this study, the latent approach was used as this process best describes the approach adopted. For an example of how coding was applied to interview transcripts, see Appendix B. All coded transcripts are available upon request.

1.5.4.3 Themes

One of the main objectives during qualitative data analysis is the task of identifying themes (Ryan & Bernard, 2003). The literature was analysed by means of latent coding, and in the process various themes were identified. The interview transcripts were analysed using the same technique (latent coding) and themes from the interviewee data were identified. A list of all identified themes can be found in Appendix C.

1.5.4.4 Categories

Various pieces of data were grouped together in order to reduce the number of different pieces of data that had to be processed in the data analysis. All similar themes were merged into one, and a record was kept of how many interviewees expressed a certain view. This number of interviewees expressing a certain view was used as a weighting, and value was assigned to themes based on this weighting. Categories are discussed in section 3.6.4 and section 4.11.

1.6 Delineation

The study is limited to one organisation in Cape Town, Western Cape. It is an in-depth case study, and the results are not generalisable in any way. The study excludes mobile devices (smart phones, tablets and laptops) and firmware upgrades of hardware, although these are discussed briefly for the sake of completeness.

1.7 Ethical considerations

The Oxford Online Dictionary defines ethics as the entire field of moral science. To address ethical issues that may arise during the research process, an open, honest and fair approach was followed. According to Neuman (2011), most ethical issues involve the balance of the value of the quest of scientific knowledge and the rights of the subjects being studied. All participants involved were made aware of the research objectives and aims. In order to avoid a conflict of interest between the researcher and the research participants, a proviso was added to the ‘permission to do research’ document. The condition was stipulated by the organisation and states that the organisation remains anonymous at all times within the thesis. This document is not included in the thesis, but is available on request.

Permission was obtained from the participants and all interviews were recorded. Participants were informed that the researcher, academic supervisor, academic co-supervisor and transcription analyst will listen to the recordings. Permission would be obtained from the participant first in the event that an external person requests access to the recordings. All recordings were encrypted using encryption software and stored on Google Drive.

All company documents obtained in electronic form were encrypted and stored in the same manner. In the event that digital copies of documents were not available, printed versions were analysed and either returned or shredded once the analysis was complete. All confidential documents were kept in a locked file cabinet while being analysed, with only the researcher able to access these documents.

The research consisted of a single organisation as the case study. The aim and objectives of the research were presented to the organisation and formal permission was obtained from the Chief Information Officer (CIO) of the company before conducting the research. The organisation is referred to as 'the company' in order to maintain anonymity. The following ethical principles were applied to the research:

Informed consent: Before the interview, participants were presented with a full information sheet containing all ethical issues relevant to the study. They were thereafter presented with the consent forms. Participants were also made aware that they could change their mind at any time, even if consent forms had already been signed.

No pressure on individuals to participate: No incentives were provided to persuade participants to participate.

Respect of individual autonomy: Participants maintained their freedom to decide what to do. Even though they signed consent forms, they could change their mind at any stage, without any explanation required.

Avoid causing harm: The research should not cause conflict and animosity among competitors and colleagues. The research process was transparent, open and honest.

Maintain anonymity and confidentiality: The identity of the organisation is anonymised. A realistic degree of anonymity is promised, according to the level afforded by the researcher. The participants' identity and contribution should be protected.

1.8 Chapter summary

The nature of the research assumes that there is no single truth or understanding of how patches are currently managed within organisations. The research methodology in this study required the gathering of data from both primary sources and secondary sources. The setting of this research is a single case study that focuses on the integration of software patch management and enterprise data security. The unit of analysis for this study was the patch management process within the company.

Convenience sampling was used to select the case and purposive sampling was used to select interviewees. Snowball sampling was utilised to identify participants who were not identified in the literature. Interviews were used and an inductive research approach was followed.

In this chapter, the topic of software patch management and enterprise data security was introduced. Some background to the problem was given and the research problem statement was identified and defined. The research questions and sub-questions were also formulated. The two main research questions are:

1) What causes the management of software patches to be complex?
2) How do organisations implement patch management in order to enhance enterprise data security?

A brief discussion of the research methodology is included, and is covered in more detail in Chapter Three. The research utilised a single case study and employed a qualitative approach.

The delineation of the study confines the research to a Cape Town-based organisation and therefore the results are not generalisable. In this chapter, the aims and objectives of the research were discussed and all ethical matters were identified and discussed.

The next chapter presents a discussion of the literature related to the research problem. Topics such as concepts and terminology relating to patch management and enterprise data security, network threats, software life cycles and software management tools will be discussed.

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

This chapter reviews some of the literature available on the topics of software patch management and enterprise data security. The literature review was conducted from a data and software perspective. The literature was reviewed using keywords from the problem statement, title, aim of the study and the research questions. This was an iterative approach as new relevant keywords were identified during the literature searches. Several data bases from the eLibrary of Cape Peninsula University of Technology (CPUT) were explored, such as Emerald, EBSCOhost, Google Scholar, ProQuest, Scopus, and Science direct. The literature was used in an iterative way, to confirm the research questions, problem statement and aim of the study.

2.2 Context of the literature review

According to Bilge and Dumitras (2012), the time between the detection of a vulnerability and the emergence of a corresponding patch is getting shorter; in some instances it is as short as a few hours. This creates pressure on IT departments to deploy patches as rapidly as possible, which is in direct conflict with best practices in terms of configuration management and assurance testing standards.

Many organisations are struggling to keep up with the constant release of software patches. Business is demanding that IT provides 100% uptime on the availability of servers, making it difficult to introduce patch deployments and other maintenance activities. The IT departments thus have to develop a process that ensures the availability of resources and the installation of security patches without breaking existing working systems (Jenkins, Arnaud, Thompson, Yau & Wright, 2014).

In many organisations, the responsibility for maintaining the server hardware reside with one team, but the applications running on these servers is supported by a different team. In cases like these, it is vital to adhere to proper change management procedures. Each server should have a change control document, and this document should state the purpose of the server, the primary and secondary person of contact, and special comments regarding the server, as well as a detailed disaster recovery plan (Chow & On Ha, 2009).

Patch management is often a difficult task, as organisations may have multiple software platforms and configurations to consider at any given time (Thakare & Gore, 2014). Gerace and Cavusoglu (2009) state that, generally, most internet assaults exploit vulnerabilities caused by misconfiguration or outdated software. They claim that the time between discovering and patching is often the problem. They further state that the frequency of zero-day attacks on organisations is on the rise, meaning the task for both vendors and organisations becomes more challenging. Gerace and Cavusoglu (2009) briefly discuss an optimal patching policy and propose a mathematical model for the trade-off between system confidentiality and availability. Using this model, they calculate the optimal frequency of the regular and irregular patching.

Arora, Krishnan, Telang and Yang (2010) argue that most cyber-attacks exploit software defects. The time lapse between initial disclosure of a vulnerability and release should thus be reduced. Sabahi (2011) states that computer security can be achieved by means of three processes: threat prevention, detection and response. The above processes can be implemented by using various components and policies such as user account access control and cryptography. This control can be used to encrypt data as well as system files.

2.3 Confidentiality, integrity and availability of data

The confidentiality, integrity and availability of data (CIA) are the most critical factors to consider when an organisation relies on information technology (IT) to manage its data (Torres, Sarriegi, Santos & Serrano, 2006). According to Sommerville (2011), three main types of threats exist within any networked environment:

i) **Threats to the confidentiality of the system and its data:** This involves the disclosure of information to people or systems that are not authorised to have access to the data (Sommerville, 2011). Confidentiality is therefore concerned with preventing unauthorised disclosure of data. Encryption is the most common method to ensure confidentiality. Access controls help provide confidentiality by restricting access to data. Steganography can also assist, by hiding data within data (Gibson, 2014).

ii) **Threats to the integrity of the system and its data:** Threats to integrity can corrupt or damage systems and its data (Sommerville, 2011). Integrity provides the assurance that data has not been modified, either intentionally or unintentionally. Hashing algorithms such as MD5 and SHA-1 can help to verify the integrity of integrity (Gibson, 2014).

iii) **Threats to the availability of the system and its data:** Availability ensures data is available when it is required. Redundancy and fault tolerance techniques such as load balancing and server redundancies can be used to increase availability (Gibson, 2014). Threats to availability can deny access to systems and data for authorised users (Sommerville, 2011). Threats are discussed in further detail in the following sections.

2.4 Threats, risk and vulnerability (TRV)

Threat, risk and vulnerability are the commonly used words in data security to describe how the CIA of data can be compromised. Microsoft developed the STRIDE model that is widely accepted. This model is developed to characterise known threats to the type of exploits being used. According to Scandariato, Wuyts and Joosen (2014:5) the STRIDE model categorises threats as follow:

> "Spoofing refers to a rogue person or program successfully impersonating another legitimate user or program. Tampering refers to a threat agent illegitimately modifying application resources, such as in memory data. Repudiation refers to a user (legitimate or malicious) able to deny the execution of an action within the system. Information disclosure refers to a threat agent obtaining private information he/she is not supposed to access. Denial of service refers to a threat agent making a system resource unavailable to its intended users. Elevation of privilege refers to a threat agent obtaining privileged access to resources that are normally protected".

2.4.1 Threats

A threat is defined as "circumstances that have the potential to cause loss or harm" (Sommerville, 2011:303). Below is a brief summary of commonly encountered computer threats:

2.4.1.1 Virus

Microsoft describes computer viruses as small software applications that are designed and coded to spread from one computer to another with the intention to interfere with computer operations. This is consistent with the definition provided by Gollman (2011:178) that a computer virus is "a piece of self-replicating code attached to some other piece of code, with a payload". The payload in this context is generally harmful.

2.4.1.2 Eavesdropping

Eavesdropping is defined as the act of listening to a private conversation on a network (Rass, 2014). Applications such as Carnivore and NarusInsight are examples of this type of software (Capriz, 2011). Social engineering is described as the art of manipulating users to disclose information such as passwords (Long, 2011).

2.4.1.3 Worm

Kurose and Ross (2010:83) describe a computer worm as “malware that can enter a device without any explicit user interaction”. Gerace and Cavusoglu (2009) discuss the slammer attack that hit in 2003 and infected tens of thousands of computers in less than ten minutes. In response to this outbreak, the security industry started asking the following questions: Why did it happen? How could we have prevented it? How can we ensure that it never happens again? Gerace and Cavusoglu (2009:117) assert that “95% of security breaches could have been prevented by keeping systems up-to-date with necessary patches”.

2.4.1.4 Backdoor

A backdoor can be a method of bypassing normal authentication, connecting to a PC, obtaining access to data, yet remaining undetected. A backdoor may be in the form of an installed program, or a modification to an existing program or even hardware (Kermani, Zhang, Raghunathan & Jha, 2013).

2.4.1.5 Direct attack

Direct access attacks occur when someone has physical access to a computer. Various types of malware can be installed and large quantities of data can be loaded onto removable storage. One way to counter this type of attack is to encrypt storage media and to keep the key separate from the system (Checkoway, McCoy, Kantor, Anderson, Shacham, Savage, Koscher et al., 2011). An indirect attack is one that is launched by a third party computer.

2.4.1.6 Denial of Service (DOS) attack

Denials of Service attacks are designed to render a network unusable. A simple example would be to re-enter a user password three times, and in doing so, locking out the account. Distributed Denial of Service attacks involve a large number of hosts that are used as a botnet. A botnet is a collection of compromised hosts that are infected with malware, and remotely controlled. Malicious users can use botnets to execute targeted DOS attacks (Kurose & Ross, 2010). The target system would then be flooded with network requests (Ehlert, Geneiatakis & Magedanz, 2010). When the attack is coordinated from multiple hosts, it is known as a Distributed Denial of Service attack.

2.4.2 Risks

A risk is defined as the "potential harm that may arise from some current process or from some future event" (Elky, 2006:1). Foley (2009) proposes an internal controls approach to manage security risks. The proposed approach could provide information about security controls that are used to mitigate risks on business processes. Security governance is responsible to prioritise and manage risks across an organisation. Security governance goes beyond the technical-centric view, and includes security in the context of the organisation as a whole (Patnayakuni & Patnayakuni, 2014).

Enterprise Risk Management (ERM) is typically used by organisations to manage risks related to their businesses. Committee Of Sponsoring Organisations – ERM (COSO-ERM) is an example of an ERM framework for documentation of the various relationships between business processes, as well as the risk and mitigating controls in place for these risks (Beasley, Clune & Hermanson, 2005). Cobit is another example of a risk management framework that focuses on various best practices for managing information systems.

ERM can be characterised by four elements. The first element is the identification of all relevant business processes. Each process consists of a number of sub-processes. The second element is the assessment of risks. A risk is described as an ambiguity in a process that could adversely impact business operations. The third element is the control element. A control is described as a framework for management of activities meant to thwart a business process from occurring. The last element is testing (Arena, Arnaboldi & Azzone, 2011).

2.4.3 Vulnerabilities

According to Kouns and Minoli (2011), a vulnerability can be described as a weakness in a system that could potentially lead to the exploitation of the system by a malicious user. Software vulnerabilities have evolved to an ongoing and serious concern because of the associated risk of exploitation by malicious users.

According to Farahmand, Navathe, Enslow and Sharp (2003), vulnerabilities describe weaknesses in a system. A vulnerability can be caused by a flaw in software (Okamura et al., 2009). Liu, Shi, Cai and Li (2012:152) state that a software vulnerability is an "instance of a mistake in the specification, development or configuration of software such that its execution can violate the explicit or implicit security policy". A security patch is designed, tested and deployed to correct these weaknesses.

Bilge and Dumitras (2012) claim that security patches are the primary method for addressing security vulnerabilities in applications. This definition is consistent with the definition provided by Shahriar and Zulkernine (2012:2), which states that a vulnerability is a "flaw in software that allows attackers to expose, disrupt, and destroy information". An exploit is defined as "a piece of software, or a sequence of commands, or even data, that can take advantage of a 'bug' in order to do harm to a system" (Begel, Khoo & Zimmermann, 2010). Vulnerabilities can therefore be exploited if it is disclosed and not patched in a timely manner.

It is crucial for software developers and vendors to make patches available as soon as possible to avoid possible exploitation (Arora et al., 2010). However, security attacks, such as viruses and Denial of Service (DOS) attacks, are in most cases a result of software vulnerabilities that have been exploited. Gerace and Cavusoglu (2009) confirm the statement made by Arora et al. (2010), claiming that software vulnerabilities are one of the main causes for security attacks. Program code is likely to contain vulnerabilities that could be exploited, intentionally or accidentally, to cause security attacks (Gerace & Cavusoglu, 2009).

Computer systems are susceptible to known and unknown attacks. Handling unknown attacks is more difficult, owing to their irregular nature (Albanese, Jajodia, Singhal & Wang, 2013). Sharma, Kalbarczyk, Barlow and Iyer (2011) highlight the fact that although program vulnerabilities have been addressed for a good number of years, security breaches are still regularly observed due to vulnerabilities being exploited. Fossi, Egan, Haley, Johnson, Mack, Adams, Blackbird et al. (2011) observe that a large number of vulnerabilities are often exploited after patches have been released. This implies that patches are not being applied, even though it is available.

2.5 Zero-day vulnerabilities

There are many ways, at many levels, that CIA of data might be compromised. One key threat worth discussing in detail is zero-day vulnerabilities and attacks. McQueen et al. (2009:1) define zero-day vulnerability as "any vulnerability, in deployed software, that has been discovered by at least one person but has not yet been publicly announced or patched". Albanese et al. (2013:3) define zero-day vulnerability as "unknown (zero-day) vulnerabilities, which even developers are not aware of". Zero-day vulnerability is therefore any vulnerability that has been deployed and has been discovered by at least one entity. Zero-day attacks exploit vulnerabilities that have not yet been disclosed on a public platform, making it difficult to analyse such threats because data is often not obtainable until after an attack is detected. Furthermore, these types of attacks are presumed to be rare events that are unlikely to be detected in experimental laboratories (Bilge & Dumitras, 2012).

According to Bilge and Dumitras (2012), there is very little defence against a zero-day attack. As long as the vulnerability remains unidentified, the affected software cannot be patched. Albanese et al. (2013) explain that traditional efforts generally assign a numeric value to vulnerability based on the likelihood that the vulnerability could be exploited. These assumptions are based on known details about each vulnerability. This approach would not be effective against zero-day attacks, because of the lack of knowledge.

Bilge and Dumitras (2012) assert that the race between zero-day attacks and the remediation measures introduced by the security industry may go on for years. In some cases, vendors may discover vulnerabilities before they are exploited or disclosed. A disclosed vulnerability that remains unpatched creates the opportunity for a cyber-criminal to create exploits and to conduct attacks on a larger scale. Zero-day vulnerabilities cannot be measured because of the less predictable nature of how these flaws are introduced in software and how they are discovered. Wang, Jajodia, Singhal and Noel (2010) address the previously stated limitation by proposing a security metric for zero-day vulnerabilities, defined as Zero-Day Safety. Wang et al. (2010) assume the existence of a complete attack graph, although in practice attack graphs for large networks are usually unfeasible.

Intrusion Detection Systems (IDSs) can help in defending against zero day attacks. The primary purpose for IDSs is to determine whether network connectivity is normal or not. The logic behind IDS is that it analyses network connections and compares them to known attack signatures. Zero-day attacks are considered the crucial challenge in the security industry. Most studies that address zero-day attacks employ unsupervised anomaly techniques to discover new attacks. A zero-day attack, by definition, has no previously known attack signature (AlEroud & Karabatis, 2012). Contextual anomaly is defined as the process of identifying patterns in data that do not conform to expected behaviour (Gogoi, Bhattacharyya, Borah & Kalita, 2011), and is used to detect events with the same attack context.

According to Bilge and Dumitras (2012), a typical zero-day attack lasts on average 312 days. They further claim that some attacks may remain unknown for up to 2.5 years. Immediately after disclosure, the volume of attacks increases by 5 orders of magnitude. Bilge and Dumitras (2012) then compared the reaction time of Microsoft and Apple to newly disclosed vulnerabilities. The findings were as follow: both vendors had un-patched vulnerabilities 180 days after public disclosure and half of all Windows users were vulnerable to 297 vulnerabilities in one year; patches were available for 65% of these vulnerabilities at the time it was first publicly disclosed.

Bilge and Dumitras (2012) propose an automated method for finding zero-day attacks in field data. The proposed method allows for the measurement of the duration of zero-day attacks. The average duration according to their research is approximately 10 months. Full disclosure policy is based on the idea that disclosing vulnerabilities to the public, instead of the vendor, is the most effective approach, as it provides an enticement for vendors to release patch faster, instead of relying on security through obscurity. Davidson (2013) defines it as trying to keep security holes a secret. Choi, Fershtman and Gandal (2010) assert that the problem with full public disclosure is that after vulnerability is disclosed, it causes an increase in attacks.

2.6 The complexities of software

According to Felmetsger, Cavedon, Kruegel and Vigna (2010), it is a well-known fact that most programs contain vulnerabilities and that these vulnerabilities may be exploited not only intentionally but also unintentionally, which would also cause security breaches.

Vulnerabilities may exist within software for various reasons, and if they are triggered intentionally or unintentionally, it can lead to a compromise of information security. A resource can be physical or logical. It may also have one or more vulnerabilities that can be exploited and, as a result, can compromise the availability, integrity or confidentiality of a resource (French, 2012).

Highsmith (2013) claims that software complexity is strongly linked to software maintenance efforts. Software complexity is described as a phenomenon that generally refers to the attributes of the data structure and procedures within software applications that may have an impact on the difficulty of understanding and changing them.

2.6.1 Software evolution

Panzica La Manna (2011) explains that generally when a system needs to be patched, the system would be shut down, updated and restarted. The problem with this approach is that a large number of critical element systems cannot be interrupted, and therefore the need for dynamic patching, also referred to as online patching, may be valid. Payer and Gross (2013) support this notion by claiming that in most cases it is only a small part of the system that needs to be updated. The goal of their research is to promote and support software development by introducing online updating of components in a distributed system. The aim of their paper is to define the conditions under which a component can be safely updated online. Panzica La Manna (2011) claims that a common theme of these works is the selection of proper time points. The system should be stable and ready for accepting a user-specified state transformation. This then results in a different state from which the system is able to continue.

Software systems are continuously changing. Changes could occur in the surrounding area of the application or in the external components. The initial requirements may change to simply to improve the quality of service. Changes may also be required when software defects are discovered (Panzica La Manna, 2011).

Lehman’s laws provide a concise summary of the software evolution process and dynamics (Sommerville, 2011). The laws state that system maintenance is inevitable as new requirements continue to emerge. With on-going maintenance the complexity of systems is likely to increase, resulting in large program evolution. The laws further state that the quality of systems is likely to decline unless systems are modified to reflect the changes in their operational environments.

2.6.2 Legacy systems

Legacy systems are old systems that are still in use, and often provide business critical functions; they can therefore not easily be decommissioned or replaced. These systems may be implemented using outdated programming languages and technology, or may rely on systems that are too expensive to maintain. It is often costly to replace these systems and documentation as it is often out of date or non-existent. Vendors generally do not support these systems anymore, leaving them vulnerable to exploitation (Sommerville, 2011).

The following is a good example of a problem with legacy systems: In 2014, Microsoft ended the support for Windows XP. Windows XP is one of the most popular operating systems and is still used even after Microsoft has stopped supporting it, making it probably the most vulnerable operating system at the moment. Windows XP can be considered a legacy system (Tutt, 2014).

2.7 Software management methods

2.7.1 Configuration management methods

The automation supplied by the configuration management system’s greatest benefit is the ability to manage multiple devices with as little effort as possible. The majority of vulnerabilities can be addressed by publicly released patches. The difficulty often lies in maintaining proper patch levels on a number of different servers in computing environments. A configuration management system reduces the burden of this portion of the system administration process (Farwick, Agreiter, Breu, Ryll, Voges & Hanschke, 2011).

Most configuration management systems add some form of complexity to an environment. The primary function of the system should be to reduce workload and simplify maintenance issues, and to streamline software and patch installations. It may be that staff require training to master these tools. It is important to note that these types of systems do not replace the need for application testing. Testing remains a necessary step before deployment, and may vary from one organisation to another, depending on size and application sets. If the configuration management is poorly defined, it could lead to more work (Chen, Mao, Mao & Van der Merwe, 2010).

2.7.2 Deployment methods

The application of patches in large environments can be a very laborious process. Insufficient time is often listed as one of the main reasons why organisations do not install patches. In some instances, system administrators simply ‘forgot’ about certain systems. These issues can be addressed with the use of configuration management systems. This, however, requires each device to have a client installed on it. One of the major benefits is that the system administrator does not literally have to remember each device on the network. As long as the clients are installed and working on the devices, patches, installations and confirmation changes can be made automatically from a centralised console (Sethanandha, 2011).

2.8 Software patches

Many definitions for a software patch exist, for example: Souppaya and Scarfone (2013:2) define a patch as a piece of software that “corrects security and functionality problems in software applications and firmware”. Although patches can serve other purposes such as the addition of new functionality, it is most often used to mitigate software vulnerabilities.

A security patch is a small software program that fixes software bugs causing security vulnerabilities to end-users. A vulnerability can be dealt with by either installing the applicable patch or removing the vulnerable software completely (Le Goues, Nguyen, Forrest & Weimer, 2012). The definition provided by Souppaya and Scarfone (2013) is used in this study, as it best describes a software patch. Patches can either be official patches released from the vendor, or third party or unofficial patches released by entities other than the vendor. An unofficial patch is described as a non-commercial patch for commercial software. It is common for security specialists to create unofficial patches if the software creators are taking too long to publish the official patch (Sotirov, 2006). A hot fix is a single package that holds information used to address a specific problem in a software product. The risk of applying a hot fix should always be considered together with the risk of not applying it (Nicastro, 2011). Prioritising which patch to install and when to install the patch is closely related. The importance of the vulnerable system should be considered, as well as the severity of the vulnerability. Dependencies that patches may have are also something to consider. It may be that the installation of one patch requires another patch to be installed first, or that the system needs to be rebooted first (Souppaya & Scarfone, 2013). The level of concern of the CIA of a company’s data depends on the criticality of the data and associated systems in the eyes of the company.

2.9 Patch management

The patch update process is a complex and costly process for large organisations. IT decision-makers often face the challenge of cutting costs while simultaneously preventing computer systems from being compromised.

According to Arora et al. (2010), a vulnerability can be discovered by multiple parties and can therefore incur different responses. The party discovering the vulnerability may choose not to publicly or privately disclose the vulnerability to the vendor or a malicious user.

2.10 Software patch management and related risks

According to Cárdenas, Amin, Lin, Huang, Huang and Sastry (2011), a risk assessment should be completed for all servers on the network. This assessment should ideally include critical data stored on servers, impact of downtime that can be introduced by the server, and the general vulnerability of the server in terms of internal and external attacks. Risk management also has an effect on the decision whether to apply patches or not. Instead of blindly deploying every single patch, a process should be followed that indicates the criticality and applicability of the software patch. Risk management and patch management can often overlap. Furthermore, the risk decision of applying the patch should be considered along with the risk of not patching the reported vulnerability.

A policy that is implemented should be based on a risk assessment, as this could potentially reduce the information security risk to a tolerable level, and in doing so, ensure that information security is addressed continuously during the lifecycle of the system. A good patch management plan should consist of various phases; risk analysis and mitigation strategies, deployment and automated tools, and repeatable defined processes (Cárdenas et al., 2011). Each server should be marked to indicate its criticality to the organisation. A higher rating should equal higher priority. These mission critical states would then relate to a risk level to the organisation. This risk factor would play an important role when deciding when and how to apply patches (Yang, Zeng, Ayachitula & Puri, 2011).

Once the organisation has been base-lined in terms of software versions, a test environment should be built. At minimum, the test environment should have minor test servers representing all mission critical applications. With IT budgets being as tight as they are, it is often not feasible to have a reasonable test environment. In these cases, patches should be deployed to the least critical servers in production first.

The risk imposed by unknown vulnerabilities is often considered as indefinable, mainly due to the less foreseeable nature of security faults. This scenario poses a major difficulty for security metrics, because a more secure configuration would be of little value if it were vulnerable to zero-day attacks (Wang et al., 2010). Souppaya and Scarfone (2013) assert that patch management tools themselves can create a security risk for originations if they are not managed properly. An organisation should strive to balance the security needs with that of availability or usability. This raises the fundamental issue that if a system is too secure or complicated to use, users may try to avoid using the system at all.

2.10.1 The importance of patch management

Patch management can often be the difference between a company running a successful IT division and an inefficient IT division. Cavusoglu et al. (2008:657) state that “today most security incidents are caused by flaws in software, called vulnerabilities”, highlighting the importance of patch management.

Mohammadi (2013) indicates that companies, government and various other institutions rely on patch management to secure their workstation against being compromised, more specifically to secure their data. They claim that patch management is one of the main components of IT security management and that it has become a key requisite in every IT infrastructure to defend networked resources from exploitation, and also to maintain a secure network. It is important to ensure that the latest patches are installed at all times across an entire computer system. Subashini and Kavitha (2011) claim that patching is done to fix security and critical vulnerabilities, but also to avoid compromising integrity and confidentiality of data.

It has been proven many times that faulty or non-existent patch management policies can result in systems being compromised. The main reason for this is the slow adoption of configuration management applications and methodologies. In recent times, configuration management systems are more widely accepted, even welcomed (Ioannidis et al., 2012).

2.10.2 Patch automation

According to Ramaswamy, Bratus, Smith and Locasto (2010), there is often a trade-off between perceived platform stability and automated software patching. Although acquiring and deploying the latest patching is considered common knowledge, it is often ignored in practice. Ramaswamy et al. (2010) further suggest that the reasons underlying the hesitancy to apply patches should be considered, and conclude that the current mechanisms of the patch deployment process may be a stumbling block. Bilge and Dumitras (2012) have a similar view that there may be a hesitancy to deploy patches when it becomes available, by saying that users often do not deploy the patches immediately for the following two reasons: the overhead associated with patch management and the general perception that patches themselves contain bugs and vulnerabilities, and potentially break a working system. Joshi (2013) confirms the notion that systems are not always patched on a regular basis because of user apathy, as well as inefficient security controls. A large number of systems remain under threat, even after vulnerability has been published and patches have been released.

Zhao et al. (2009) propose an approach that will enable the automation of both the notification and rectification phase. This approach also offers some flexibility that enables an administrator to customise according to specific requirements, regardless of the vendor of the products involved. The proposed solution has five main components. First, the email client is responsible for receiving incoming advisories from various vendors. Automated notification is responsible for reading incoming advisories and then storing the advisories in a profile if the vendors' digital signature has been verified. Advisory databases store all the incoming advisories, along with classification data. Relevant advisories are those relevant to the local system and are displayed to the system administrator. Automated rectification acts on the relevant advisories by downloading a specific patch from the vendor's website and then consulting the local patch policy for accepting them. Patches are then stored in a repository allocated for this purpose. After subscribing to relevant lists with the correct filtering criteria, it becomes easy for administrators to build a system database of relevant advisories. The main advantage of this approach is that it enables advice from multiple sources to be viewed through a common interface.

Jansen (2011) proposes the deployment of a virtual machine server. Huang, Baset, Tang, Gupta, Sudhan, Feroze, Garg et al. (2012) explain that the server is responsible for communicating with the vendor website, downloading and analysing the patch, storing relevant information to a database, distributing patch files to a client agent and then constructing information regarding the present status of the patch installation for the client. The patch client agent has the responsibility of communicating with the patch server, scanning the system's information, detecting the installed and uninstalled patches in a system and installing patches where required.

Wei, Lu, Jafari, Skare and Rohde (2011) outline the set-up of an automated vulnerability solution and present the details of a prototype system implementation. This system would verify incoming alerts, process messages from several vendors and prioritise this within a common management interface. Giuffrida, Kuijsten and Tanenbaum (2013) focus on the automation of live updating for major operating system upgrades without rebooting or shutting down, and his proposed solution is based on the idea of state latency, a phenomenon that allows updates to occur only in predefined system states. The solution can routinely perform transactional live updates at the process level, improving the overall update process.

Allowing the user to manually install patches is not a good idea for a number of reasons. Typically, the users receive a popup saying that a patch can be downloaded. The user may dismiss this popup (leaving the computer in a vulnerable state), or multiple users can download this patch simultaneously causing congestion on the network. There is very little, if any, benefit to allowing a user to perform this function (Souppaya & Scarfone, 2013). There is also a downside to bundling patches. The time between when a vulnerability is exposed and when it is patched is increased. The attacker thus has a greater window of opportunity to exploit the vulnerability because of the deliberate delay in the release of the patch with the next bundle, instead of immediately. If it becomes clear that exploitation is occurring, a vendor may release software patches faster.

2.10.3 Hot patching

Hot patching is described as the process of applying patches without shutting down or restarting a system (Ramaswamy et al., 2010). This phenomenon is useful for systems that cannot be unavailable for any reason (Gonzalez & Locasto, 2013). Ramaswamy et al. (2010) investigated the phenomenon of hot patching and concluded that offline patching remains the predominant form of patching at present. They believe there is value in a reliable hot patching solution as it will reduce downtime. Their study focuses on the pros and cons of the phenomenon. They explain that mission critical systems are often the hardest to patch as they cannot afford downtime, or administrators may fear instability or downtime as a result of the patch. Inaction in this case may hold as much risk as proactive action. Here the risk of patching should be weighed against the risk of not patching. Their research aims to make hot patching possible and less risky.

2.10.4 Horisontal patching

Stolikj, Cuijpers and Lukkien (2013) conducted an experiment with two test cases and confirmed that horisontal patching can be used to improve the patching process in large networks of devices that share a common software component. Horisontal patching is defined as a method for handling code differences in systems running several applications on top of common software.

Wright (2014) also reports on the phenomenon of horisontal patching. Wright defines it as a method for optimising the size of incremental updates in a multi-application environment, where dissimilar devices share a common software component. Wright (2014) argues that horisontal patching gives better results as the number of heterogonous devices grows. He further adds that the speed comes with the trade-off of extra processing time, possibly for a number of options to grow exponentially.

2.10.5 Patch management configuration

Another challenge for organisations is the fact that there are numerous methods by which patches can be deployed. Souppaya and Scarfone (2013) identified six methods by which software patches can be installed:

i) A software product may be able to automatically update itself
ii) A centralised operating system management tool may be able to initiate a patch deployment
iii) A third-party application may be able to initiate a patch deployment
iv) Network access controls and health checks may be able to initiate patching
v) A user may manually direct software to automatically update itself
vi) A user may install software patches manually

Souppaya and Scarfone (2013) also say that having several methods of deploying patches can cause conflict. It can be challenging when multiple methods are trying to install the same patch, especially when the organisation explicitly does not want to deploy certain patches. Patches may also be overlooked because one system administrator may assume another administrator has taken care of a particular patch.

2.10.6 Software patch management challenges

Inadequate application inventory management procedures could introduce a challenge, because patch management is reliant on current inventory information of the applications that is installed on every device in the environment at any given time (Souppaya & Scarfone, 2013). The high rate of vulnerability disclosures has placed the focus and attention mainly on practical matters and immediate matters such as vulnerability disclosure, the speed of patch development, the dissemination of patches and the application thereof. While these issues are important, often more subtle issues are ignored. One such issue is the consideration of how many vulnerabilities are already in existence that have been discovered by malicious users, but have not yet been publicly disclosed (McQueen et al., 2009). The primary contribution of their work is to propose a method for making sound estimates on the amount of zero-day vulnerabilities in existence on any given day.

Shahriar and Zulkernine (2012) argue that the problem of patch management should be addressed at the root. The aforementioned argument was substantiated with a comparative analysis of various vulnerability mitigation studies that were done in the past, and found that the mapping between various techniques and the vulnerabilities and limitations they address may be somewhat obscure. According to Shahriar and Zulkernine (2012), roughly 50% of all security vulnerabilities occur at application code level, and although application vulnerabilities have been addressed by academia for more than twenty years, security breaches still occur regularly. This indicates that the problem has not yet been solved.

According to Okamura et al. (2009), computer security has been recognised as one of the most important issues within the domain of software reliability engineering, and one effective countermeasure is to remove the flaws causing these security problems in the software. Naik and Tripathy (2011) claim that high-quality software testing leads to high-quality secure systems. For this to be true, it requires that as many faults are removed from software in the testing phase as possible. This is especially important for proprietary software. This should be tested extensively before it is shipped to the market, since the developer is withholding the source code.

Challenges that complicate patch management often lead to compromises in security that could easily have been prevented. If organisations can minimise the time spent on patching activities, they can utilise that time to improve other security efforts. Many organisations have largely operationalised their patch management efforts, with the aim of making it a core information technology (IT) function as opposed to a function that is part of security (Souppaya & Scarfone, 2013). McNaughton, Ray and Lewis (2010) state that release management is a service operation phased in service computing literature. McNaughton et al. (2010) claim that few studies have considered release management from the Information Technology Infrastructure Library (ITIL) viewpoint compared to other ITIL processes.

ITIL release management defines the mechanism and strategies for developing and releasing software and hardware, and is a module of the Service support set in ITIL version 2. Cobit and Accredit Solutions for information can be used in conjunction with ITIL process. This could be used to define business and operational goals, metrics, and roles for patch management. Both Cobit and ITIL consist of best practices at a generic level and suggest what should be considered in the process implementation (Sihvonen & Jäntti, 2010).

2.10.7 Release management and patch management

According to Lahtela, Jäntti and Kaukola (2010), ITIL is a widely used service management framework. It consists of various best practices for service delivery and is an auditable standard for IT service management. It consists of two parts. The first component is the specification for service management and the second is the code of practice for service management.

Lahtela et al. (2010) claim that patch management relates to ITIL release management, configuration management, and change management. Sihvonen and Jäntti (2010) claim that most research examines patch management as a technologically intensive process, although it is largely affected by human influences. It requires skilled resources and commitment from management, well defined process and roles, and enabled - that would be the technology.

Sihvonen and Jäntti (2010) look at the implementation of release and patch management processes and state that patch management is a sub-process of release management within the software maintenance. Their study focuses on the type of challenges that are related to the release management and patch management process and makes recommendation for how these can be avoided.

Lahtela and Jäntti (2011) state that a configuration management system can increase the capabilities of system administrators to effectively deal with patch releases and upgrades, on both the operating system and application level, as well as help with the management of system configuration across any environment, regardless of the size.

2.11 The role of software vendors in the patch management process

Both IT companies and consumers need a well-defined and effective patch management process to deploy hardware and software release packages. The IT company is responsible for designing, building, testing and deploying the package. The IT customer assumes the responsibility of receiving these software patches and installing them (Sihvonen & Jäntti, 2010).

Singhal and Ou (2011) conducted a study on the severity impact of the number of attacks, and determined that application vendors tend to devote more effort to develop patches for vulnerabilities that have a more severe impact, than they do for vulnerabilities having a less severe impact.

Implemented programs often contain vulnerabilities that might be exploited to cause security breaches, as most security vulnerabilities belong to program code. There is no all-inclusive analysis of different vulnerability mitigation actions. The mapping between the techniques, the address vulnerabilities and the limitation of different techniques may be obscured (Shahriar & Zulkernine, 2012).

2.11.1 Vendors' behaviour

Many vendors have predefined dates for patch releases. Microsoft, for example, releases software patches every second Tuesday of the month. It is known as 'Patch Tuesday' (Zseby, King, Brownlee & Claffy, 2013). The theoretical basis for most studies is that the quality and the release time of security patches are determined to a large extent by the economic behaviour of vendors. A vendor is more reactive if a large portion of the customer loss is internalised by the vendor. Software vendors might have an incentive to release a faulty software product, and then fix it later by means of a patch (Arora et al., 2010).

The development of a patch normally starts when a vulnerability is disclosed to the vendor. If an attacker acquires the knowledge about the vulnerability before it has been patched, a successful attack can be launched. The entire process involves various role players such as system administrators, software vendors, government agencies and attackers. The state of knowledge has a direct impact on the state of the vulnerability management (Arora et al., 2010).

Zhu, McQueen, Rieger and Basar (2011) claim that the timing between the discovery of vulnerability and the availability of a patch for the vulnerability become crucial for the assessment of the security risk exposure of software users. According to Souppaya and Scarfone (2013), product vendors have countered this conflict by improving the overall quality of patches, as well as bundling patches for their product. This means, instead of a vendor releasing a large number of patches over a period of time, the vendor simply releases their patch in a single bundle once every quarter, for example. This directly reduces the need to prioritise patches, as the organisation now only need to prioritise the bundle instead.

Schryen (2011) investigated the vulnerabilities and security patches of 17 closed- and open-source application packages. The study shows that vendor policy was the most significant factor. Vulnerabilities should be disclosed responsibly and ideally a patch should be available at the time a vulnerability is disclosed publicly (Okamura et al., 2009).

August and Tunca (2011) compared the effectiveness of three alternatives: keeping the vendor liable for damages, keeping the vendor liable for patching costs, and forcing government to impose security standards. August and Tunca (2011) argue that with the increase of zero-day attacks, vendors should be held liable to a certain extent.

Cavusoglu et al. (2008) studied the vulnerability disclosure mechanisms by means of an analytical model. They showed that although each disclosure mechanism does ensure the release of patches, early vulnerability disclosure does not necessarily lead to faster patch release by a vendor.

Beres and Griffin (2012) state that certain vendors may decide to release patches in a bundle once every cycle, where a cycle can be monthly, quarterly or even yearly. An example of this is a patch release by Oracle for the month of October. This patch contained 127 new bug fixes (Oracle, 2013). Ransbotham and Mitra (2013) state that if vulnerabilities are publicly disclosed without the vendor responding in due time, it can create a window of opportunity for malicious users.

The development of patches incur an expense for the software vendors, and also the pressure of delivering a patch fast might result in poorly developed patches that can introduces a new set of problems. It is often the case that certain patches introduce new security vulnerabilities or system instability. Furthermore, if a user perceives the application of patches as tedious and risky, users may skip the patch process all together (Okamura et al., 2009).

2.11.2 Disclosure

A vulnerability may or may not be exploited, purely depending on who discovers it. A vulnerability remains a zero-day vulnerability until it is reported to the vendor, or publicly announced, whichever comes first. The disclosure policy ultimately affects the quality and speed of patch development (Wright, 2014). Zhu et al. (2011:51) propose compartmentalising “the task of vulnerability management into different submodules: discovery, disclosure, development and patching”.

Bilge and Dumitras (2012) suggest that there may be value in disclosing new vulnerabilities to the public, even if patches are not available yet. When disclosing vulnerabilities to vendors, it can be done responsibly or irresponsibly. Publicly announcing the vulnerability without giving the vendor an opportunity to create a patch is considered irresponsible. Responsible disclosure would involve reporting the vulnerability directly to the vendor, giving them a fair chance to develop a patch before it is publicly announced (Arora, Telang & Xu, 2008). The vendor however can decide what to do with the information. It may develop a patch, or it may keep the vulnerability a secret in the hope that obscurity solves the problem (McQueen et al., 2009).

Mell and Scarfone (2007) studied the disclosure threat effect that can be described as the effect on patch release time in the possibility that another vendor releases a patch first, and inadvertently discloses the vulnerability. Bilge and Dumitras (2012) indicate that to mitigate the risk of disclosure, all vulnerable hosts should be patched as soon as the patches become available. Arora et al. (2010) and Cavusoglu et al. (2008) analysed the impact of full disclosure and they could not find any strong evidence to suggest that patches would immediately be released following disclosure.

Schryen (2011) considers the topic of vulnerability disclosure. He claims that public disclosure can increase the risks posed by security attacks. He explains that publicly disclosing information could expose the systems of unprotected users to the possibility of attacks. Arora et al. (2010) argue that whilst public disclosure might have some disadvantages, it can also be an effective method to accelerate the release of software patches by vendors.

2.11.3 Initiatives and incentives

Most vulnerability researchers can now get paid for their research effort. Hewlett Packard's (HP) Zero-Day initiative pays a substantial bounty for vulnerabilities in enterprise software products. In 2013, the initiative accepted 290 vulnerabilities (Bilge & Dumitras, 2012). Szefer, Keller, Lee and Rexford (2011) state that in addition to focusing on fixing known vulnerabilities, organisations should focus on closing potential attack vectors. Several vendors have incentivised the approach of vulnerability discovery. HP for example, has the zero-day initiative that pays developers up to R80 000 for any reported exploits. Security firms also run similar initiatives.

2.12 Considerations for developing a patch management strategy

2.12.1 Sources of patch information

Joshi (2013) indicates that various sources provide patches, information about patches, as well as vulnerability information. These repositories include government websites, technical blogs, security bulletins, software vendors, among others. By their nature, these sources are unavailable and unused by automated vulnerability management systems.

Most of the time, the internet is the first reference for information on software vulnerabilities, exploits and attacks. More specifically, the information is found in text at security bulletins, vulnerabilities databases, news reports, blogs and various chat rooms and forums. Joshi (2013) explains that the unstructured nature of this text is a limitation on the automation of vulnerability management, and presents an automatic framework that enables the generation and publication of a Resource Description Framework-linked data resource for vulnerability descriptions and other concepts across the internet. Enterprise patch management is reliant on having a current and comprehensive inventory list of all software that is earmarked for patching on each host. This inventory should be as detailed as possible and should include which software is installed to each host, and which version of the software is installed.

Keeping current with patches can be a challenging task. It is important to have current information that will allow for quick decision-making with regard to patch management. This could help with deciding which patches are critical, which are merely ‘nice to have’, and which are unnecessary (Le Goues, Forrest & Weimer, 2013). Joshi (2013) claims that the collaboration of various information sources in a machine understandable format can possibly prevent zero-day attacks. The Linked Data Principles describe various web standards such as HTTP, Uniform Resource Identifier (URI) and Resource Description Framework (RDF). These standards help identify access, describe and interlink data on various sources (Joshi, 2013).

In order for an administrator to have a proper awareness of the problem, administrators have to monitor numerous sources of information, as no single reliable source is available (Joshi, 2013). Organisations running systems from multiple vendors need to monitor the advisories of these vendors, typically by subscribing to their organisation’s mailing lists or newsletters. Although some vendors have taken the approach to automatically alert users, this only provides a partial solution to the problem.

2.12.2 Roles and responsibilities

Roles and responsibilities need to be defined on an individual basis, and the roles need to be explained from beginning to end to avoid confusion. A policy should state who is responsible for monitoring and testing as well as patch deployment alternatives. A monitoring team has a very critical function and their performance can be the difference between a successful and ineffective patching strategy. Effective monitoring will allow for patches to be tested as soon as they become available to the public (Souppaya & Scarfone, 2013).

2.12.3 Vulnerability management

The process of rectifying vulnerabilities consists of the following: the correct patch needs to be downloaded, tested and distributed for end-effected systems. Although in the past these tasks were undertaken manually by system administrators, it is no longer feasible to have these tasks undertaken manually. With the increase of the number of vulnerabilities, and also the rate at which they are exploited, doing it manually is not a sustainable approach anymore (Rinard, 2011).

Patch management forms part of a bigger and more broadly defined problem called vulnerability management. Effective patch management should be a systematic and repeatable process. Patching is necessary for security, but it is often difficult to manage systematically. It is often the case where conflicting priorities have to be balanced in order to minimise disruption to mission-critical systems (Liu et al., 2012).

Zhao et al. (2009) indicate that to achieve effective vulnerability management, the system administrator should receive current information pertaining to new vulnerabilities. Furthermore, this should be followed-up by applying any updates that is relevant to their systems. If patches are not planned properly, it can easily cause overload on the network if all devices try to download the same patch at the same time. This can be addressed by staging the process in phases, relative to the capabilities of the infrastructure such as network throughput and server capacity (Souppaya & Scarfone, 2013).

2.12.3.1 Patch classification systems

The Common Vulnerability Scoring System (CVSS) and the Common Weakness Scoring System (CWSS) both measure the relative severity of vulnerability. This measurement is done in isolation and does address their overall impact. The measurement systems also provide security professionals and vendors with a standard method for assigning numerical scores to known vulnerabilities which are listed in a public vulnerability database (Mell, Scarfone & Romanosky, 2007).

Adobe has rating systems, called the Adobe Rating System. According to Schryen (2011), this system serves as a guideline for customers with managed environments. The priority ranking is based on historical attack patterns, the type of vulnerability, and the platform and potential mitigations that may be in place. According to Adobe, a Priority 1 is defined as an update that resolves a vulnerability being targeted, or that may have a higher risk of being targeted. Adobe recommends Priority 1 patches be installed with 72 hours of its release. A Priority 2 is defined as an update that resolves vulnerabilities that have been at elevated risk in the past. Their recommended time for patching this vulnerability is 30 days. A Priority 3 patch is defined as an update fixing vulnerabilities that have not been a target for attack in the past. Adobe recommends that customers install this patch at their own discretion (Schryen, 2011). Adobe also rates the severity of vulnerabilities according to the following scale: Critical, Important, Moderate and Low. A critical vulnerability is one that, if exploited, would allow native code to execute, potentially without a user being aware (Mell, Scarfone & Romanosky, 2006).

Souppaya and Scarfone (2013) assert that for a patch management policy to be effective, the CEO or owner of the organisation should approve the policy. Furthermore, the policy should provide guidelines for the responsibilities, testing and deployment procedures. The patch management program should be included in the policy. Wang et al. (2010) propose a network security metric called k-zero-day safety. They believe it is impossible to measure which unknown vulnerabilities are likely to exist, and thus start with the worst-case scenario. Their metrics then count how many zero-day vulnerabilities would be required to compromise a network node. The larger the overall count, the more secure a network, which means the likelihood of having more unknown vulnerabilities at the same time, on the same network, should be lower.

Wang et al. (2010) claim to have validated this claim by applying it to the evaluation of existing practices in network hardening through a series of case studies. This metric would enable a direct measurement and comparison of the various security solutions in place. Existing metrics typically assign a numeric score to a vulnerability based on various attributes and known facts about the vulnerability. With that being said, this is not feasible with zero-day vulnerabilities as no previous information is known about the vulnerability. The metrics proposed by Wang et al. (2010) consider the time and effort required by potential adversaries, as referenced in the Markov model of various attack stages. Qian, Mao, Rayes and Jaffe (2011) claim that the length of the shortest attack patch in terms of the number of exploits and conditions is taken as a security metric for the measurement of the amount for security of networks.

2.12.4 Testing

In the past, Microsoft and various other vendors have been known to release patches that caused other problems such as blue screen errors or unexpected system reboots. Testing is often considered the difference between a good patching program and a company losing millions in revenue as a result of downtime. It is wise to perform a phased approach when rolling out any software to a production environment, regardless of what testing has been done in laboratories and virtual environments. Deployment to the production environment always holds some form of risk, but this risk can be minimised (Mell, Bergeron & Henning, 2005).

Naik and Tripathy (2011) emphasise the importance of testing. If vulnerabilities are discovered in the testing phase, they can be fixed before the software is released. McQueen et al. (2009) argue that the early discovery of vulnerabilities does not implicitly lead to them being patched. It is occasionally believed to be more expensive to rewrite the code and is not considered cost effective.

One of the biggest challenges in patch management is the process of testing patches before they are deployed. With too little testing you run the risk of unexpected behaviour in the production environment, but with too much testing, you increase the windows of exposure, thus increasing the risk of the exploitation of vulnerabilities. Furthermore, it remains critical to test patches before they are deployed to ensure the stability of new patches in the current environment. Organisations might not have adequate testing facilities available, and it can contribute to the fact that this function is often omitted. Organisations may lack the personnel needed to test patches (Le Goues et al., 2013).

Ideally, an organisation would deploy all new patches immediately as it is released to minimise the time that systems are susceptible to specific weaknesses. However, in reality, this is seldom possible because organisations have constrained resources, which call for patches to be prioritised in order of importance. There is a significant risk in installing patches that have not been tested, as they could result in operational disruptions and potentially do more harm than good. Souppaya and

Scarfone (2013) acknowledge that testing patches is a time-consuming task and places even more strain on company resources. In patch management, prioritisation, timing and testing are often in conflict.

After patches have been tested and are ready to be distributed, the system custodians should document proposed changes to the system, as well as results. If any incidents occur, they should be documented along with the solution that resolved the issue. The testing of mission critical or business applications should be done during office hours in case something goes wrong so that disaster recovery tasks can easily be implemented. The maintenance window should always allow enough time for a possible rollback (Souppaya & Scarfone, 2013).

VMware provides a cost-effective method to build a test laboratory. This method, however, only allows for OS compatibility testing and does not account for hardware variables at all. Although this method of testing is incomplete, it is still better than having no test environment at all. A full backup is vital before the deployment of any patches to production environments (Yu, Han, Schneider, Hine & Versteeg, 2012).

Deployment procedures rely heavily on the monitoring and testing guidelines that are established by the patch management policy. Some companies still do patch installation manually, and although there are benefits to this method, at some point in time, the cost could outweigh the benefits (Souppaya & Scarfone, 2013).

2.13 Chapter summary

This chapter reviewed the literature from a software and data security perspective. It became evident that the user community often does not apply patches immediately for a variety of reasons. Some of the reasons include that fact that patch management is likely to add some form of complexity to an environment. The importance of a patch management policy was noted, as patch management forms a critical part of the overall security of an organisation. Infrastructural limitations such as available bandwidth were considered, and should be kept in mind when designing a patch management policy. All risks related to patch management should be identified upfront, as this can add to the complexity of patch management at a later stage.

Different software platforms require different approaches toward patch management. For example, high availability systems such as emergency contact centres and online baking systems poses a challenge to patch as they generally cannot be rebooted or shut down without significant consideration and a valid reason. A link between patch management and data security was noted, as it appears that an unpatched environment is more likely to be compromised.

Software vendors typically do not support legacy applications and this poses a risk for organisations that rely on legacy applications and systems. Windows XP is a well-known example of this scenario where non-vendor supported software remains in use, despite there being no official software patches released for it.

The importance of testing emerged, specifically the challenges around a test environment and the limitation of resources. Organisations often have constrained resources, and patch management may not be correctly prioritised. There is no centralised source of patch information. This implies that organisations have to subscribe to different vendors' newsletters and websites for patch releases and information.

The following chapter will discuss the research philosophy, the research approach, data collection and ethical consideration in greater detail.

CHAPTER THREE: RESEARCH DESIGN AND METHODOLOGY

3.1 Introduction

According to Welman, Kruger and Mitchell (2005), research methodology considers the logic behind the research methods applied, as well as the research process and the decisions that led to those processes being followed.

This chapter discusses the chosen research methodology and design. The research philosophy and research strategy are discussed. Various data collection and analysis methods are considered, and ethical considerations are presented. The chapter concludes with a summary of the research methodology.

The objective of the study is to identify some of the challenges organisations face when implementing patch management and to determine how patch management affects enterprise data security, and the challenges faced by organisations with multi-vendor systems.

The aim of this study is to explore the challenges preventing organisations from proactively implementing patch management. The problem is viewed from the perspective of the participants by means of interviews with key role players. Awareness of the context in which the research occurs is maintained at all times. The empirical basis of this case study is an organisation located in Cape Town. Convenience sampling was used to select the case. Neuman (2011) describes convenience sampling as sampling that has availability and convenience as the main criteria for selection.

Fossey, Harvey, McDermott and Davidson (2002) state that it could be beneficial for the researcher to assume an evolving understanding, and then design the research strategy accordingly. The afore-mentioned approach allows for some flexibility in the research. Noor (2008) claims that the choice of which research methodology to employ is largely dependent on the complexity of the research problem. Morgan and Smircich (1980) support this notion and assert that the actual suitability of the methodology is derived from the nature of the phenomena to be studied.

According to Noor (2008), there are two methodological traditions within social science research: positivism and post-positivism. Noor (2008) explains positivism as an approach to the construction of knowledge by means of research that emphasises the model of natural science. Post-positivism is described as a methodology where a reality is socially constructed, instead of it being objectively determined. The task of the social scientist should not be solely to gather facts and establish how often certain events occur, but should rather be to appreciate the different constructions and meanings that people place upon their experience.

According to Yin (2013), a typical assumption is that the research report should present the data and then carry out the analysis. This is a linear sequence and mimics the reporting of most quantitative research methods. The choice of data collection procedures should be guided by the research questions and the choice of design. The case study approach often combines data collection methods such as archival research, interviews, questionnaires and observation. The choice of the data collection method is also subject to constraints of time, finances and access (Meyer, 2001).

Eisenhardt (1989) states that focusing on a single case forces the researcher to pay special attention to the case. On the other hand, a multiple case study might help to strengthen the findings from the entire study. This is consistent with the view of Yin (2013), asserting that the researcher should decide whether or not to use theory development to help select the case and develop a data collection protocol and organise initial data analysis strategies. Gerring (2004) is of the same view that a good case study involves defining the case and justifying the choice for single or multiple cases. According to Yin (2013), the goal of the researcher is to choose a good case study and to collect, present and analyse data fairly. For the purpose of this study, a single case study was conducted.

3.2 Research philosophy

3.2.1 Ontology

Ontology is described as a system of belief that reflects an understanding of an individual about what constitutes a fact, and reflects "the belief in a pre-existing empirical reality" (Hirschheim & Klein, 1989:1207). Neuman (2011) describes ontology as unambiguous specifications of a shared conceptualisation. It thus contains the type of object or concept that exists, as well as their containing properties and relations.

Ontology is concerned with the question of whether social entities need to be observed as subjective or objective. Subjectivism and objectivism and can therefore be seen as two important components of ontology. Objectivism is an ontological view that social phenomena and their meanings have an existence that is independent of social actors.

The nature of this research assumes no one single truth or understanding about how organisations currently manage their patch deployment process (subjectivism). Real-life conditions were explained in the context in which it exists, and in the process objective knowledge is produced. Therefore a subjectivism approach was best suited and chosen for this research.

3.2.2 Epistemology

3.2.2.1 Interpretivist

Interpretivism integrates human interest in the study and is usually focused on meaning, and may employ multiple methods in order to gain insight in different aspects of the issue. Interpretivism is an ontological position which asserts that social phenomena and their meaning are continually being accomplished by social actors.

Within the interpretivist paradigm, findings and knowledge is created as the research progresses. The findings emerge through dialogue in which interpretations are negotiated. Pragmatic and moral concerns are important, therefore sufficient consideration was given to ethical issues. Furthermore, all interpretations are based in a particular context and time, and always remain open to reinterpretation and negotiation. Therefore, this research follows an interpretivist stance.

3.2.2.2 Positivist

According to Welman et al. (2005), the positivist seeks to expose general laws of relationships, which considers all the people all of the time. Research design can be viewed as the blueprint for achieving a research objective and answering the research questions. The process of selecting the research design can be challenging, due to the number of methods available (Blumberg, Cooper & Schindler, 2005). Bearing this in mind, the researcher decided to use a single case study method, and interviews with relevant participants were conducted.

3.3 Research approach

Johnson and Onwuegbuzie (2004) define qualitative research as a type of scientific research that seeks to answer a question by gathering proof, producing findings and using systematic approaches. In addition, qualitative research seeks to understand a given research problem from the perspective of the affected research population.

According to Bryman (2006), research methods are identified in the pursuit of obtaining answers to the research questions. By doing so the researcher is able to address the research problem in a logical manner by using facts that can be verified. With this in mind various types of research methods were utilised. These included interviews as well as findings and ideas from previous studies.

Qualitative research allows for the in-depth analysis of problems and produces descriptive data. An example would be someone's opinion in an interview. Qualitative research seeks to gain an insider's view of the phenomena under investigation by probing participants to share their experiences. Interviews were conducted with a selected group of participants who are involved in the current patch management process, and information was obtained, such as current challenges, recommendations and past experiences.

Qualitative research collects information that is typically not in numerical form; therefore, qualitative data is harder to investigate than quantitative data because it deals with different views, opinions, feelings and experiences. The quantitative researcher views the world as predicable and being of a single truth. The nature of this research is empirical and the research has taken an objective approach towards the research problem. Due to the nature of the research a quantitative research approach was adopted.

Quantitative research methods are primarily concerned with techniques that analyse numbers, or with the results of numerical processes. Quantitative methods may include laboratory experiments, surveys, numerical methods and formal methods. Qualitative research methods as those methods that generate data that is not in numeric form, such as words and images. These methods allow the researcher to capture what the participants wish to express, as opposed to having them tick pre-determined responses.

Rich and detailed descriptions of specific events were used. Given the nature of the research, a qualitative approach was adopted. Neuman (2011) describes qualitative research as an approach that is concerned with analysing, recording, and attempting to uncover deeper meaning in things such as human behaviour and experience. These may include opposing behaviours, beliefs and emotions.

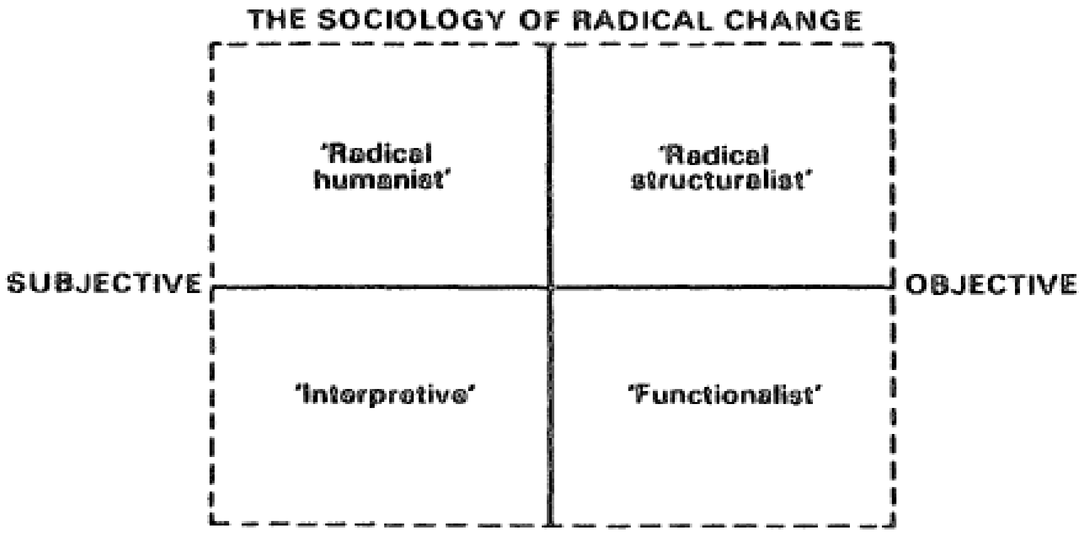


Figure 3.1: Four paradigms for the analysis of social theory
(Source: Burrell & Morgan, 1979:22)

Burrell and Morgan (1979) present four research paradigms as depicted in figure 3.1. According to Cronjé (2012), in order to achieve each of the four research paradigms, two questions should be answered. The first research question is: "What causes the management of software patches to be complex?" Through attempting to answer this question, the current generic and specific challenges facing organisations with regard to patch management became evident. The first research question highlights the importance of patching, and the consequences of failing to manage the patch process efficiently. The second research question is: "How do organisations implement patch management in order to enhance enterprise data security?" This research question highlights the components that should go into a patch management policy. This question will also explain the benefits an organisation can obtain from an integrated patch management framework. A single in-depth case study (as opposed to a surface-level study of numerous cases) was conducted.

Holloway and Wheeler (2013) indicate that qualitative research enables a holistic perspective. Qualitative research should thus assume that a whole phenomenon is under investigation. Ritchie, Lewis, Nicholls and Ormston (2013) add that qualitative research typically incorporates an emergent design. The whole research design process cannot be determined upfront. Further understanding has evolved as the research progressed, and certain data collection and analysis techniques have led to further collection and analysis activities.

A qualitative approach is applicable for this study because it explores a phenomenon. The research instruments are flexible and the research methods allow for the use of focus groups, semi-structured interviews and participant observation (Cohen & Crabtree, 2006). In this study, Interviews and the literature was used. Qualitative research is descriptive as it focuses on describing and understanding a phenomenon, and therefore is best suited to this study. Descriptions may include, but are not limited to, the context, participants, activities and processes (Holloway & Wheeler, 2013).

This research is concerned with "what" and "how". The research questions are: "What causes the management of software patches to be complex?" and "How do organisations implement patch management in order to enhance enterprise data security?" According to Denzin and Lincoln (2000), qualitative research also involves fieldwork. This implies direct contact with participants involved in the phenomenon in the natural setting of the phenomenon.

According to Kitzinger (1994), qualitative methods allow for greater freedom in terms of the interaction between the participant and the researcher. Open-ended questions were used most frequently. According to Kawulich (2005), open-ended questions have the ability to produce meaningful data that may have been unanticipated by the researcher. The data is often considered rich and exploratory. Denzin and Lincoln (2000) explain that qualitative research implies an importance on processes and meanings that do not involve a rigorous examination process; it is often not measured in terms of quantity, frequency or intensity. This method is for researchers who are interested in insights, discovery and interpretation rather than the testing of a hypothesis.

Benbasat, Goldstein and Mead (1987) state that in the design phase of case study research, the researcher should define the case that is being studied. Yin (2013) claims that a virtue of using case studies is the ability to redefine the "case" after early data collection. This could, however, require the research to backtrack in some ways (for example, reviewing slightly different literature and perhaps reconsidering the initial research questions). The researcher came to the conclusion that a qualitative approach would be best for describing and understanding the processes and activities involved in the patch management process.

3.3.1 Inductive

Qualitative research acts on inductive logic with no pre-determined frameworks, and the aim is to improve understanding of human behaviour. Inductive research produces descriptions of how and why people do what they do, and enables the researcher to advance their understanding of a particular phenomenon or process. The research therefore does not gather data to support preconceived hypotheses or theories.

In an inductive approach, a researcher begins by gathering data that is applicable to the research topic. Once adequate data have been collected, the researcher looks for patterns in the data. In an inductive approach, it is common practice for the researcher to start with a set of observations and then move from specific experiences to more general propositions about those experiences. This research followed an inductive research approach.

3.3.2 Deductive

Quantitative research acts on deductive logic and the conceptual framework is clear; it seeks to substantiate what is right or wrong with human behaviour. It is common for a deductive approach to start with a hypothesis, while an inductive approach generally starts with research questions in order to guide the scope of the study. With inductive approaches the aim is generally focused on exploring a new phenomenon or looking at previous studies from a different viewpoint. Inductive approaches are normally associated with qualitative research, while deductive approaches are usually associated with quantitative research. The deductive approach starts with a social theory.

3.4 Research strategy

3.4.1 Case study

Eisenhardt (1989:534) defines a case study as “a research strategy which focuses on understanding the dynamics present within single settings”. According to Yin (1981), an exploratory case study focuses on “what” questions. The setting of this study is a single case study focusing on the integration of software patch management and enterprise data security. Yin (2013:13) defines a case study as an “empirical enquiry that investigates a contemporary phenomenon within its real-life context”. The boundaries between the phenomenon and the context may not be clearly evident.

A single case study is described as a variant of a case study that includes one observation of the phenomenon (Stake, 2013). The phenomenon is investigated within the company across three different domains. The issue of patch management was considered from a head-office perspective, a stores perspective and a server perspective.

According to Stake (2013), a case study can be useful when the opportunity to learn is considered important. A case study provides a method of enquiry for in-depth investigation of a phenomenon; hence it is the method of choice for this study. Given that patch management is a complex technical and managerial process that is comprised of various activities, participants and processes, the case study method was well justified for this study.

The case study employed various methods of data collection and analysis, such as interviews, official company documentation reviews and literature review. The primary source of information was interviews with relevant participants. Stake (2013) highlights the importance of setting boundaries for the case. A time boundary should be set indicating the beginning and end of the case. The authors maintain that setting boundaries for the case is also known as the process of conceptualisation.

Noor (2008) states that a case study is not intended to study a complete organisation, but rather to focus on a specific issue, or unit of analysis. In this research, the researcher focuses specifically on the issue of patch management and how it affects enterprise data security. In order to examine and understand the patch management process, the researcher has decided to do an in-depth single case study. This method enabled the researcher to understand the complexities of patch management process activities as they are experienced in a real-life context. The disadvantage of case studies is that they are not scientifically rigorous in their methodology and do not address the issue of generalisability. The author further states that a case study approach also has its strengths, in that it enables the researcher to gain a complete view of a certain phenomenon, and it can provide a complete picture as many sources of evidence can be used. Shavelson, Phillips, Towne and Feuer (2003) assert that the case study method is vital when the research aim is to address a descriptive question such as “what” or an explanatory question such as “how” and “why.” The case study method assisted the researcher to make direct observations and allowed for data to be collected in a natural setting, as opposed to relying on derived data. Gummesson (2007) argues that one advantage of a case study is the opportunity for a holistic view of the process.

Meyer (2001) claims that a case study is a loose design that requires that several design choices be made. The first consideration is to decide how many cases to include. The cases should be sampled and the researcher then decides on a unit of analysis. The second consideration is to decide if it should be a single or multiple case studies. Meyer (2001) further states that in a case study, the researcher has the choice of an all-inclusive design or an embedded design. While a holistic design examines the global nature of the phenomenon, an embedded design focuses on sub-units. This research adopted an embedded design.

The case study method is often used to add to knowledge of an organisation, group or individual, or of a social or political phenomenon. This method is often used in disciplines such as sociology, psychology, political science, social work, community planning and business. The need for case study research arises mainly out of the need to understand complex social phenomena. The case study method allows the researcher to investigate and retain a holistic view of actual events.

According to Yin (2013), “what” and “how” questions are more explanatory and are likely to lead to the use of case studies, histories or experiments as the research strategy of choice. “What” and “how” questions are the focus of this study. According to Cronjé (2012), the research questions posed by the researcher determine what quadrant the research resides in. Considering the research questions within this study and keeping in mind the interpretation presented by Cronjé (2012), this research falls within the interpretive domain.

A common concern with case studies is that they provide very little basis for scientific generalisability. It is often the assumption that one cannot generalise from a single case (Flyvbjerg, 2006). Yin (2013) responds to this issue by saying that case studies, similar to experiments, are generalisable to theoretical propositions, but not necessarily to populations of the universe in general. According to Yin (2013), proof for case studies may come from a variety of sources such as archival records, documents, participant-observation, interviews, direct observation and physical artefacts.

According to Flyvbjerg (2006), the case study produces context-dependant knowledge which allows individuals to develop from a rule-based beginner to a topic expert. Context-dependant knowledge and experience is at the very core of expert activities; hence the case study can also be seen as a generator of such knowledge. Flyvbjerg (2006) claims that Harvard University is among the few universities in the world that have adopted the notion of context-dependent learning, thus promoting the use of case studies as a research method. The case study method should not be used in the hope of proving anything, but rather in the hope of learning something.

This research is based on a single case study at a retail organisation. The organisation was chosen because of convenience and because the organisation is representative of the research problem.

3.5 Data collection

Bryman (2006) states that with qualitative research, the primary instrument for data collection should be the researcher. Qualitative research thus assumes that collected data is mediated directly by the researcher. Qualitative research should be inductive and focus on discovery, in order to aid in understanding. This research does not start with a hypothesis or a preliminary conceptual model. Instead various techniques were used for data collection and these will then be analysed using an inductive approach. Bearing all this in mind, a qualitative research approach was adopted for this study.

A good case study benefits from having several sources of proof. While collecting case study data, the main idea is to triangulate converging lines of evidence to make your finding as vigorous as possible. Yin (2013) states that case study research can include both qualitative and quantitative data. The evidence in a case study should be presented with adequate clarity to allow the audience to judge independently of the researcher's understanding of the facts.

Meyer (2001) indicates that when a researcher relies on interviews for primary data, the issue of trust becomes a crucial factor. During the research process, the researcher personally approached each participant and explained the research aims and objectives. This was followed up with a formal meeting request. Interviews should be of an open-ended nature through which key respondents are asked about a specific matter, and also their opinions about certain events. The respondent can suggest other persons who should be interviewed, as well as other sources of evidence (Yin, 2013). This is a method the researcher employed within the research instruments, by making last interview question the following: "Thank you for your time. Who else would you recommend I speak to?" By doing this, the researcher achieved some form of triangulation because eventually no more new names came up.

Yin (2013) indicates that using recording devices is largely subjected to personal preference, but admits that audiotapes provide a more precise version of the interview than most other methods. According to Yin (2013), recording devices should not be used in the following instances:

- When an interviewee refuses permission of the use of a voice recorder
- When there are no plans to transcribe an interview
- When the recording device itself creates a distraction in the interview
- When the recording device is considered an alternative for listening during the interview

As none of the above was the case in this study, the researcher used a recording device because there was a deliberate intention to transcribe the interviews. The researcher obtained written permission from each participant prior to the interview process. Furthermore, the recording device is small and easy to use, so no distractions were caused during the interview. By using a recording device, the researcher was able to focus all attention on asking the questions, listening to the responses, and probing the interviewee as and when required.

3.5.1 Sampling

Sampling can be seen as a cost-effective method of collecting data when the research sample is too large to study the entire sample. The size of the sample was determined by choosing representatives from each of the departments involved. The research sample is depicted in Appendix E. According to Neuman (2011) the main source of sampling is specific cases that could deepen the researcher's understanding of the phenomena being studied. In this case the phenomenon was the integration of software patch management and enterprise data security.

The case study organisation is located in Cape Town. Convenience sampling was used to select the organisation. Neuman (2011) describes convenience sampling as having availability and convenience as the main criteria for selection. The selection of interviewees was informed by the literature, and included personnel from the server team, internal audit, system administrators, security personnel and enterprise architecture. Purposive sampling was used to select interviewees. Purposive sampling is described as "the most common sampling technique", where "the researcher actively selects the most productive sample to answer the research question" (Marshall, 1996: 523). Snowball sampling was used to identify participants who were not explicitly identified in the literature, and yet were connected to the patch management process either directly or indirectly in this particular case.

Participants may not know each other, but information obtained from one may lead to the next.

Purposive sampling is described as a data collection method whereby the researcher identifies additional data required, as the study progresses and understanding emerges. According to Bock and Sergeant (2002:241), "two objectives are at the heart of purposive sampling: ensuring that all relevant types of people are included in the sample, and changing the sample structure during the research process". Purposive sampling is better described as qualitative enquiry and is based on informational considerations as opposed to statistical considerations.

Quota sampling is considered to be a type of purposive sampling. The size and characteristics of the research sample is determined upfront during the design phase. Snowball sampling is when current participants refer the researcher to other people who may possibly be able to contribute in the study (Marshall, 1996). Purposive sampling was used in this research as a sampling method. The roles of respondents were initially suggested by the literature and expanded as the research progressed.

3.5.2 Unit of analysis

Neuman (2011:69) describes unit of analysis as the "units, cases, or parts of social life that are under consideration". The unit of analysis for this study was the patch process within the organisation. The patch process was researched by means of conducting interviews and reviewing company documents such as the information security policy and the patch management policy.

3.5.3 Unit of observation

The unit of observation is the level at which data was collected. In the case of this research it was the people within the organisation who play a role in the current patch management process. Within the company, affected departments were identified: the Head Office environment, the Server environment and the Store environment. The people within these respective environments were the units of observation.

3.5.4 Interviews

Various methods can be used to conduct research, such as interviews, user observation, focus group and ethnographic interviews. For the purpose of this study, Interviews were used as the method of data collection. The author was the primary data collection instrument, and used an interview guide (see Appendix F) that lists the interview questions in the sequence it was asked of interviewees. The researcher reserved the right to ask additional questions for clarification. With the necessary consent from the participants, the interviews were recorded. Information obtained from these interviews was analysed. The recorded interviews were encrypted with Microsoft Bit Locker and stored on Google Drive. Only the researcher, academic supervisor, academic co-supervisor and transcription analyst have access to the interview recordings.

In-depth interviews are ideal for collecting data on individual people's histories, perspectives and experiences. In qualitative research, the type of data generated includes field notes, audio, video and transcripts. Interviews were selected as the primary data-gathering instrument. Interview questions were carefully designed to address all research questions and sub-questions that were not adequately addressed by the literature. The questions were piloted with a participant in the presence of the research supervisor. The purpose of the pilot interview was to identify ambiguities, clarify the wording of the questions, test the order of the questions and determine whether certain questions should be added or omitted. Information was collected with the help of various information technology professionals, who agreed to be interviewed. The interviewees included staff members from the following departments: information security, desktop support, server support, store support, internal audit and enterprise architecture.

3.5.4.1 Semi-structured interviews

All interviews were voice recorded with the written permission of the participants. The reasons for recording the interviews were to: i) maintain a high level of accuracy and richness of data; and ii) for the researcher to focus his undivided attention on asking the questions and probing the participants as and when required. All interviews were transcribed, which allowed for in-depth analysis of the data.

According to Meyer (2001), the goal of interviews is to see the research topic from the perspective of the interviewee and to understand how the participant came to have this view. To achieve this goal, King (1994:16) claims that interviews should have a "low degree of structure". These are called semi-structured interviews. The questions should be predominantly open-ended and should focus on specific situations in the world of the interviewee, as opposed to abstract and general opinions.

When structured interviews are conducted, the analysis process is relatively straightforward. All the different responses to a question can be grouped together. Semi-structured interviews are useful when the researcher is trying to explore a variety of points of view and does not want to be limited to a specific schedule (Berg & Lune, 2004). For the purpose of this research, Interviews were semi-structured. A basic set of open-ended questions was used. The researcher reserved the right to ask additional questions for clarification. Information obtained from these interviews was transcribed and analysed. Semi-structured interviews were used.

Participants were provided with a copy of the research guide before the time, in order to familiarise themselves with the interview questions, enable them to opt out should they feel they may not be able to contribute to the topic, or to choose not to participate in the research.

3.5.4.2 Interview guide

All respondents were asked the same questions, in the same sequence and no prior predictions were made in terms of possible responses. Purposive sampling was used to select participants for the interviews. The interviews played an important role in the data collection process. The research guide included open- and close-ended questions in order to gain a deeper understanding of the phenomena under investigation. The interview guide was used to direct the conversation in the direction of the topics that would be of relevance towards the research. Table 4.1 maps the research sub-questions to the corresponding interview questions (see Appendix F for a copy of the interview guide).

3.5.5 Summary of data collection

This section outlines the data collection methods and approaches adopted in the study. Data was collected in textual form from the literature and official company documentation. Company documents helped with verifying some of the findings in the interview, and initially assisted with the development of the interview guide. Primary data was collected with the interview method. Semi-structured interviews were conducted in order to maintain a certain level of flexibility. All interviews were voice recorded, with the written permission of each participant, to maintain a high level of accuracy. Recorded interviews were then transcribed for further analysis.

3.6 Data analysis

Yin (2013) asserts that analysing data can often be the most troublesome process. This is because of false expectations that the data will speak for itself. Yin (2013) refers to what he calls the “reverse” lesson; the researcher will realise that the key fundamental assumptions for later analysis are made at the beginning stages of the case study. Yin (2013) further states that case study analysis can rely on several methods, the usage of which might be expected during the initial design of the case study. The analysis can be presented during a case study as one gradually builds an argument that addresses the research questions. Furthermore, the report does not have to follow any particular form. The opportunity therefore exists to compose a case study that can be more exciting and allow for more creativity to report the findings, than with other methods.

Data was collected in textual form from the literature and interviews (transcripts). Collected data was never converted into numeric form for statistical analysis; therefore, a qualitative research approach was used. Document analysis was used to analyse organisational documents such as policy documents, change control documents and release management documents. Content analysis was used to analyse interview transcripts (Xu & Croft, 1996).

Hermeneutics refers to the theory of interpretation. Myers (1997:10) explains that “hermeneutics is primarily concerned with the meaning of a text or text-analogue”. Ideas are nested in historical, linguistic and cultural categories of meaning. It maintains that a problem can only be genuinely understood through a solid understanding of its origin. This can be better explained as the theory of understanding text. This research considered various company documents such as policies and frameworks. Hermeneutics was a useful method to analyse the content in both organisational documents and interview transcripts.

All interviews were voice recorded with the permission of the respondents to ensure an accurate account of the conversation and to prevent data loss. Recordings are stored on Google Drive, where only the research supervisor and the researcher can access the recordings. All recordings were labelled with sufficient metadata to streamline the data collection process.

Documentary evidence is useful as a method to verify information that was collected in interviews, given the fact that what people say may differ from what they do. Documents also provide guidelines for assisting the researcher with questions for the interviews. Only official company documents were used. According to Noor (2008), examining a number of sub-cases enhances the validity, accuracy and reliability of the results by capturing an all-inclusive essence of the topic that is being studied.

The researcher should be concerned with the correctness of the document. The most important use of documents is to verify evidence from other sources. The correct spelling of a name and job title is an example of the type of information that can be corroborated in organisation documents. If documentary evidence is contradictory rather than corroboratory, the researcher needs to conduct further research into the topic (Yin, 2013). Archival records were used to a small extent, and included organisational charts, as this helped with identifying the initial list of participants.

This is considered one of the most important sources of evidence in a case study. It is best practice for interviews to be guided by conversations, rather than structured queries. According to Yin (2013), the researcher has two jobs: firstly, to follow the researcher's personal line of enquiry as echoed by the case study's research methodology, and secondly, to ask interview questions in an impartial manner that also serves the original line of enquiry.

Triangulation is the rationale for using numerous sources of evidence. The use of only one source of evidence is not recommended for case studies, as one of the major strengths of the case study method lies in the opportunity to use many different sources of evidence. There are four common types of triangulation: investigator triangulation, data triangulation, methodical triangulation and theory triangulation. Data triangulation is often the most relevant one in terms of case studies. This encourages the researcher to collect information from several sources with the aim of verifying the same phenomena (Yin, 2013). According to Yin (2013), when pursuing a corroborating strategy, it is important to note that true triangulation occurs when facts about a study have been supported by more than one source of evidence.

3.6.1 Transcribing

King (1994) asserts that various studies collect data in audio and textual form. In the case of this study, data was collected in audio format during the interviews. The researcher acquired the services of a transcriptionist to transcribe the interviews. A non-disclosure agreement (see Appendix H) has been signed by the transcriptionist, as the transcripts contain the name of the organisation and the names of certain interviewees. All transcripts were verified by the researcher to ensure that it is an accurate representation of the interview recording. The process was repeated until all transcripts had been verified. For examples of three transcribed interviews, see Appendix A. All other interview transcripts are available on request.

3.6.2 Coding

Latent coding is where the researcher would look for the underlying meaning in the text (Neuman, 2011), and was used in this study. The researcher read transcripts several times in order to identify common ideas, themes and phrases. Identified statements were then highlighted as depicted (see Appendix B). A complete list of coded transcripts is available on request.

3.6.3 Thematic analysis

All identified themes that became evident during the study, were recorded. These themes provided the basis for the findings of the study and were used as input data to create the final output. The themes were first divided into two main categories: those obtained from interviews and company documents, and those obtained from the literature. These categories were then further divided into main themes and sub-themes. The themes were explored and summarised into a set of key themes, which formed the basis for the discussion of results in Chapter Five.

One of the objectives during qualitative data analysis is the task of identifying themes. The unstructured themes can be found in Appendix C and the structured themes can be found in Appendix D. Themes are discussed in more detail in section 4.10.

3.6.4 Categories

The first stage of data analysis is to eliminate errors from the collected data. After all errors have been removed, the latent coding technique was applied in this study. After this process, themes were identified and categories were coded accordingly (Vaismoradi, Turunen & Bondas, 2013).

Similar phrases and statements were grouped together to reduce the amount of data that required analysis in the data analysis process. All identified themes were initially divided into two main categories: those obtained from interviews, and those obtained from the literature. The main themes are used for categorising analysed data. These categories were then further categorised together based on similarity of statements.

3.7 Ethics

Each participant was contacted in advance to explain what the research was about and to ask if they would be interested in participating. The participants were able to withdraw at any time; even after the interviews they had the right to request that their interview be discarded. All recorded interview files and transcription documents were kept in a safe location. Interviewees were presented with a consent form (see Appendix G) that had to be signed.

According to Neuman (2011), most ethical issues involve the balance of the value of the quest of scientific knowledge, and the rights of participants being studied. All participants involved were made aware of the research objectives and aims. In order to avoid conflict of interest between the research participants and the researcher, a proviso was added to the “permission to do research” document. This point states that the organisation remains anonymous and whatever discoveries were made can only be published if permission is granted by the organisation. This condition was stipulated by the organisation.

All interviews were recorded with permission from the participant. Participants were informed that only the researcher, academic supervisor, academic co-supervisor and transcription analyst have access to the interview recordings. Permission would be obtained from the participant first in the event an external person requests access to the recordings. All recordings were encrypted using encryption software and stored on Google Drive.

The research consisted of a single organisation as the case study. The aim and objectives of the research were presented to the organisation and formal permission was obtained from the Chief Information Officer (CIO) of the company before conducting the research. The organisation is referred to as ‘the company’ in order to maintain anonymity. The following ethical principles were applied to the research:

- **Informed consent:** Before the interview, participants were presented with a full information sheet containing all ethical issues relevant to the study. Participants were thereafter presented with the consent forms. They were also made aware that they could change their mind at any time, even if consent forms had already been signed
- **No pressure on individuals to participate:** No incentives were provided to persuade participants to participate

- **Respect of individual autonomy:** Participants maintained their freedom to decide what to do. Even though they signed consent forms, they could change their mind at any time, without providing an explanation
- **Avoid causing harm:** The research should not cause conflict and animosity among competitors and colleagues. The research process was transparent, open and honest
- **Maintain anonymity and confidentiality:** The identity of the organisation is anonymised. A realistic degree of anonymity is promised, according to the level afforded by the researcher. Participants need remain in control of the disclosure of their identity and also their contribution to the research

3.8 Chapter summary

The nature of the research assumes that there is no single truth or understanding of how patches are currently managed within organisations. The research methodology in this study required the gathering of data from both primary sources and secondary sources. The setting of this research is a single case study that focuses on the integration of software patch management and enterprise data security. The unit of analysis for this study was the patch process within the organisation. The patch process was researched by means of conducting interviews and reviewing company documents such as the information security policy and the patch management policy.

The first step in the research process was to develop an area of interest and define the research topic. This was followed by a brief literature survey and various discussions with professionals in the field of information technology, as well as the research supervisor. This was useful in gaining an early understanding of the research problem that needed to be explored. After the idea was formulated and documented, it was presented and defended at a formal meeting known as the Faculty Research Community (FRC) at CPUT.

Convenience sampling was used to select the case; purposive and snowball sampling were used to select interviewees. Interviews were used and an inductive research approach was followed. The interview recordings were transcribed, and the transcripts analysed with the coding method.

The research design and methodology was discussed in this chapter. Various data collection and analysis methods were considered, and a single case study method was selected for this study. Interviews and document review was used as data collection methods. The difference between quantitative and qualitative research was explained, as well as the reason for using a qualitative approach. The ethical aspect of the study was discussed in detail, and the chapter concluded with a summary of the research design and methodology. The next chapter will present the data collection and preliminary data analysis.

CHAPTER FOUR: DATA COLLECTION AND PRELIMINARY DATA ANALYSIS

4.1 Introduction

The previous chapter described the research methodology that was adopted in this study. This chapter considers the data collection process that relied on interviews, a literature review and a document review of certain company documents. These documents included the organisation's information patch management framework and patch management policy. For the convenience of the reader, table 4.1 maps the research sub-questions to the corresponding interview questions.

Table 4.1: Research sub-questions and interview guide mapping

Research Sub-Questions	Corresponding Interview Questions
1.1 What challenges do organisations face with regard to patch management?	• How familiar are you with the generic issues in Patch management? • How familiar are you with your company's Patch Management issues?
1.2 What challenges are created by a multi-vendor environment?	• What are the problems associated with managing patches from any external vendors? • Are you are aware of any test procedure for patches before they are deployed? • How are patches prioritised within the organisation? • How are patches acquired from different vendors?
1.3 What controls can organisations implement in order to deploy the relevant patches?	• Based on past experience, what incidents occurred as a direct result of failing to properly manage patches?
2.1 How do organisations prioritise patches for deployment?	• To what extent is patch management part of your job? • What components would you like to see go into a patch management policy?
2.2 How do organisations identify patches best suited for their security needs?	• What components would you like to see go into a patch management policy? • What are the problems associated with managing patches from any external vendors?
2.3 Who are the key role players in managing the patch deployment process?	• Are you aware who the role players are in the patch management process? • Who do you think should be added or removed from the patch management process? • Who should develop the policy, implement the policy, and police the policy?

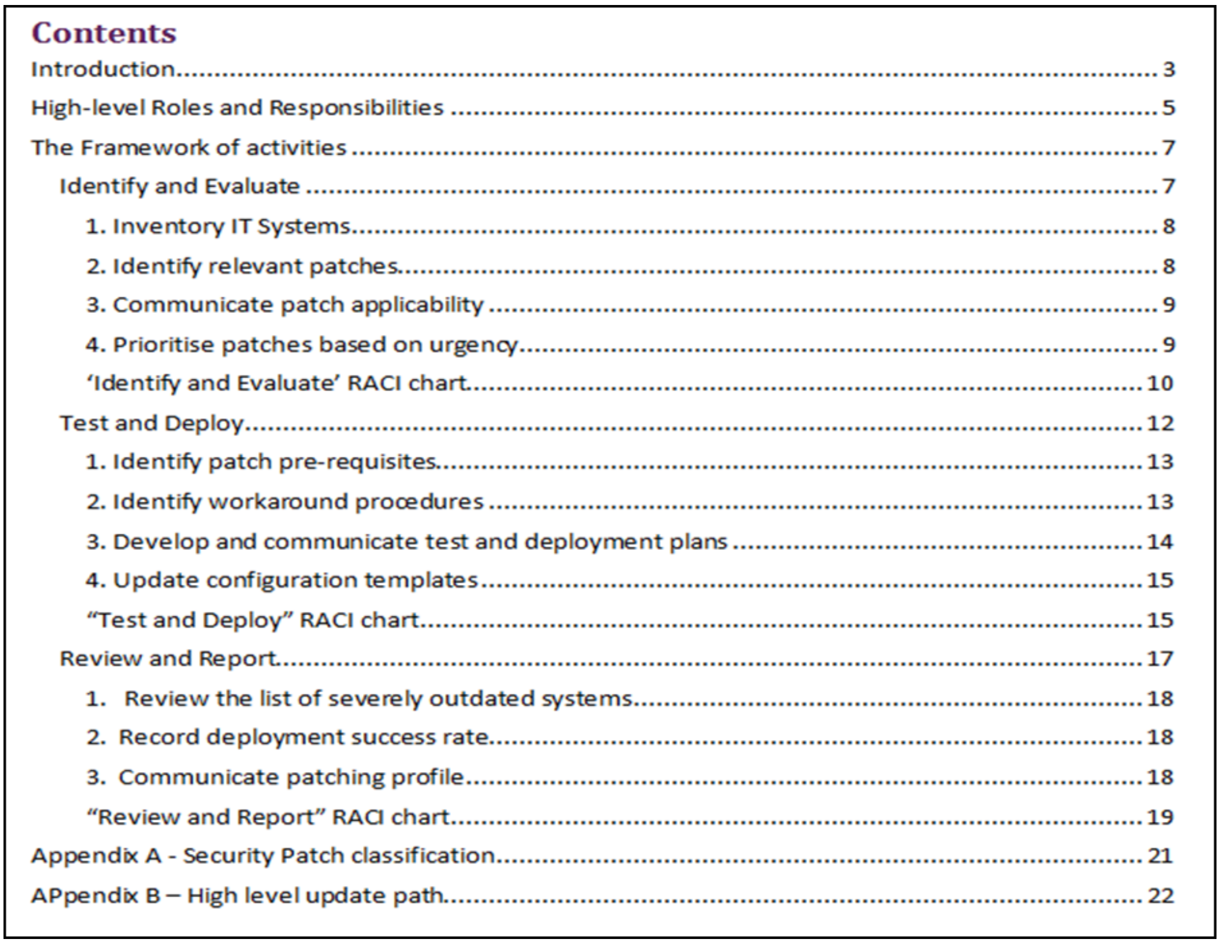
4.2 Data collection

In-depth interviews were conducted with twelve participants spread across six departments (also see section 3.4). The departments included operational teams, enterprise architecture, information security, and internal audit. For a visual representation of the interviewees and the departments they represent, see Appendix E. Each participant was either directly involved in the patch management process or represented a team that was involved in the process.

4.3 Document review

The purpose of the document review was to gain an understanding of the current patch management process and structures in place within the organisation, and to determine how familiar the various departments were with these two documents. Furthermore, the patch management policy and patch management framework were analysed using latent coding. Key themes were identified and this was primarily used as input when designing the interview questions. The interview questions addressed some of the issues that were not clear in the documents. Written permission was obtained; the table of contents of both documents are presented in figure 4.1 and figure 4.2 respectively.

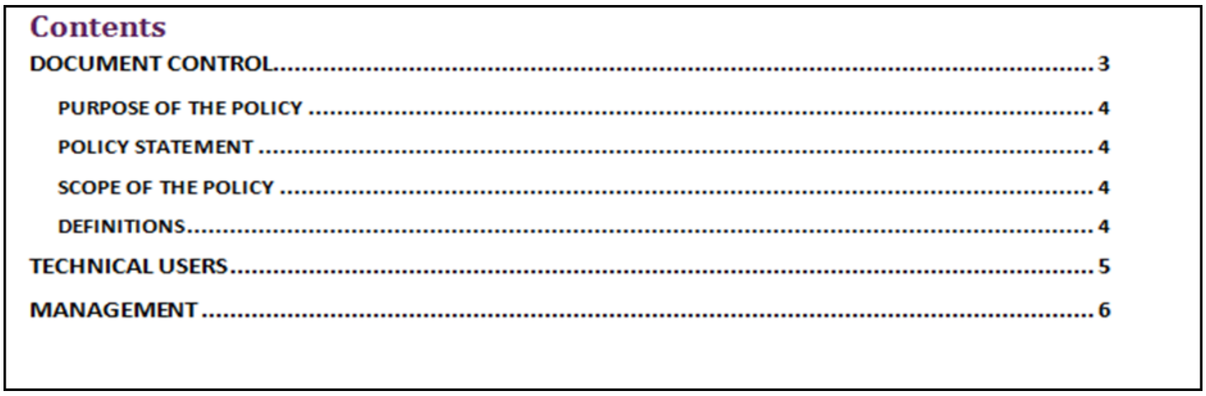
Company documents were obtained from the organisation's intranet with written permission from the CIO, but with a disclaimer. The organisation stated that the contents of the patch management framework and patch management policy documents are considered private and for internal use only. The researcher was allowed to review and analyse the documents, but not to explicitly divulge the contents thereof as it contains confidential information such as the N-strategy that would indicate to an outsider how vulnerable the organisation would be at any given time, or the exceptions criteria that would indicate to an outsider what systems are not patched at all. The organisation was anonymised and referred to only as 'the company'—this was a condition of the 'permission to do research' document.



Contents

Introduction....3
High-level Roles and Responsibilities....5
The Framework of activities....7
Identify and Evaluate....7
1. Inventory IT Systems....8
2. Identify relevant patches....8
3. Communicate patch applicability....9
4. Prioritise patches based on urgency....9
'Identify and Evaluate' RACI chart....10
Test and Deploy....12
1. Identify patch pre-requisites....13
2. Identify workaround procedures....13
3. Develop and communicate test and deployment plans....14
4. Update configuration templates....15
"Test and Deploy" RACI chart....15
Review and Report....17
1. Review the list of severely outdated systems....18
2. Record deployment success rate....18
3. Communicate patching profile....18
"Review and Report" RACI chart....19
Appendix A - Security Patch classification....21
APpendix B – High level update path....22

Figure 4.1: Table of contents: the company's patch management framework



Contents

DOCUMENT CONTROL....3
PURPOSE OF THE POLICY....4
POLICY STATEMENT....4
SCOPE OF THE POLICY....4
DEFINITIONS....4
TECHNICAL USERS....5
MANAGEMENT....6

Figure 4.2: Table of contents: the company's patch management Policy

4.4 Description of case

The company is a large organisation located in Cape Town, with an ICT footprint substantial enough to be representative of the research problem. The organisation has three main operational divisions from a patch point of view, and each division, due to their nature, has a different approach towards patch management. Figure 4.3 depicts the layout of the case at a high level.

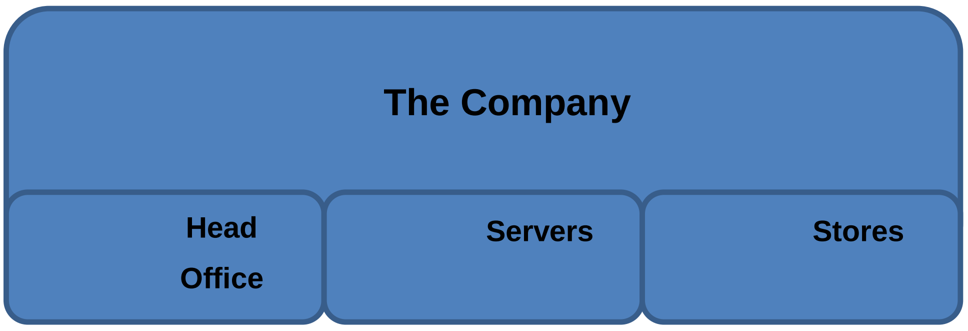


Figure 4.3: High-level layout of the case organisation

The company has more than 15000 workstations and 750 servers. The environment is segregated into a Head Office environment, a Server environment and a Store environment. Below is a brief description of what each department is responsible for:

- **Head Office**: The head office team is responsible for the head office workstations. These include windows workstations (operating systems supported: Windows XP, Windows7, Windows 8 and Windows 10) as well as Apple Mac workstations (operating systems supported: Yosemite and El Capitan). The head office operational team is responsible for, among various other functions, the patching of all head office workstations.

- **Server Support**: The server team is responsible for the entire server infrastructure within the organisation (operating systems supported: Windows Server 2003, 2008, 2012 and UNIX based operating systems). Services supported on these servers include, but are not limited to, database servers, exchange servers and web servers. The server team is responsible for the upkeep and maintenance of all servers, and patching is included within their responsibilities.

- **Store Support**: Store support is responsible for the upkeep and management of all field workstations (operating systems supported: Windows XP and Windows 7). These devices typically have bandwidth limitations when compared to head office and server infrastructure. The store support team is responsible for the maintenance of all field devices, and the patching distribution and deployment to these machines is one of their functions. These machines are geographically dispersed across Africa and rely on various technologies to connect them to the head office. Technologies include, but are not limited to, ADSL, Metro Ethernet, and wireless and fibre connections. The nature of these various technologies adds further complexity to the patch management strategy.

As can be seen from the above descriptions, each environment has a different hardware and software architecture. Therefore, it made sense to consider each environment individually in terms of how the patch process was managed.

4.5 Research sample

Table 4.2 summarises the current job titles of the thirteen participants. The table also includes the previous two job roles of each participant. The reason for displaying table 4.2 is to assure the reader that all participants have adequate experience and knowledge on the research topic and could potentially make a valid contribution.

Table 4.2: Summary of current and previous job roles of participants

Interviewee (I)	Current Job Role	Previous Job Role 1	Previous Job Role 2
1	Head Office Manager	IT Technical Specialist	Service Desk Team Leader
2	HO Technician 1	Desktop Support Analyst	Junior Programmer
3	HO Technician 2	Infrastructure Specialist	Desktop Support
4	Server Manager	Unix Administrator	Oracle Database Administrator
5	Server Technician 1	Server Engineer	Server Specialist
6	Server Technician 2	Team leader	SQL Database Administrator
7	Stores Manager	Network Manager	Team Leader
8	Stores Technician 1	Infrastructure Specialist	Server Management
9	Stores Technician 2	IT Technical Specialist	Server Engineer
10	Enterprise Architect 1	Development Architect	Programmer
11	Enterprise Architect 2	Network Manager	Programmer
12	Security Specialist	Information Security Specialist	IT Systems administrator
13	IT Auditor	Desktop Configuration Analyst	Desktop Support Analyst

4.6 Sampling

Sampling is covered extensively in Chapter Three, but for the convenience of the reader a brief summary is presented here. Convenience sampling was used to select the organisation. The selection of the role players to be interviewed in the case study was informed by the literature, and included personnel from the operational teams, internal audit, information security and enterprise architecture. (see table 4.2 for a detailed description of each participant.) Purposive sampling was used to select interviewees. Furthermore, snowball sampling was used to identify additional participants who were not explicitly identified in the literature and yet were connected to the patch management process either directly or indirectly.

4.7 Interview process

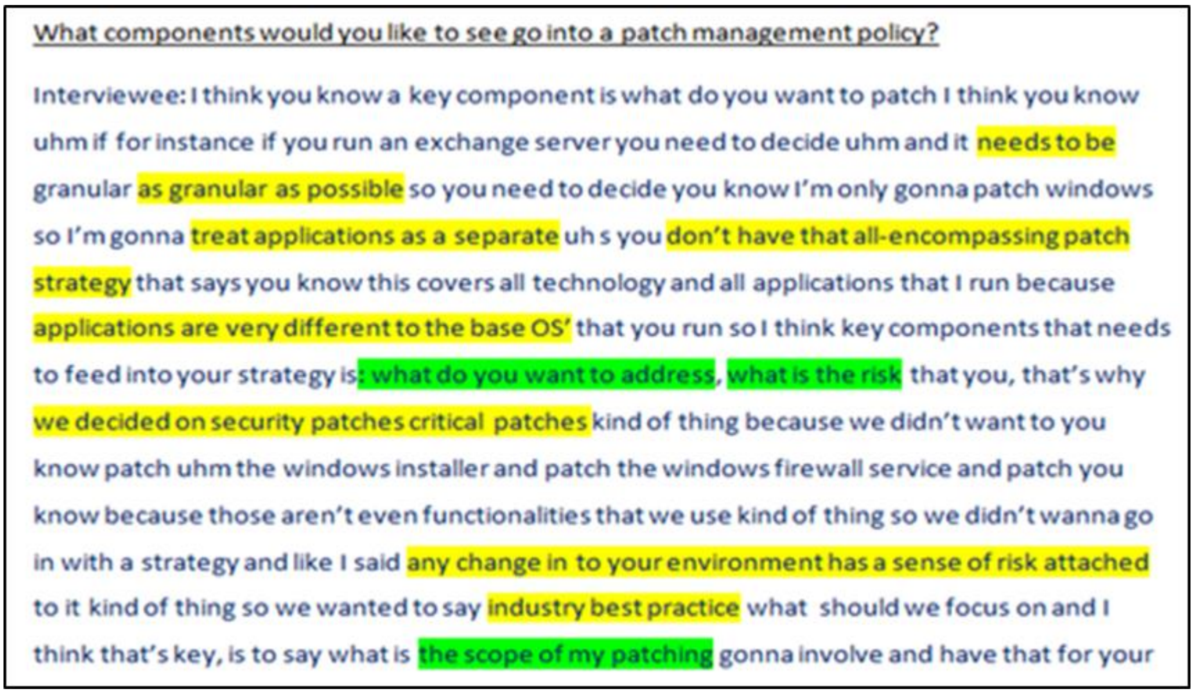
For the purpose of this study, semi-structured interviews were used as a method of data collection. The interview guide lists the interview questions in the sequence that the questions were asked (see Appendix F). With the necessary consent from the participants, the interviews were recorded. The recorded interviews were encrypted with Microsoft Bit Locker and stored on Google Drive.

4.8 Transcribing

In this study, written permission to record interviews was obtained from all interviewees. All interview recordings were transcribed, and while this function was outsourced, the researcher verified the accuracy of the transcripts afterwards. Participants' names were mentioned during the interviews, but the transcription analyst signed a non-disclosure agreement (see Appendix H), stating that none of the information in the interview transcripts may be divulged in order to adhere to ethical principles and maintain the confidentiality of the interviewees. The researcher verified all recordings against the transcripts to ensure accuracy and to become familiar with the data. For examples of three transcribed interviews, see Appendix A. All other interview transcripts are available on request.

4.9 Coding

For the purpose of this study, latent coding was applied. Similar themes were coded in the same colour and these were then grouped together in an Excel spread sheet for further analysis. The researcher looked for common phrases, but also searched for underlying meaning in the text. The latent approach was used as this process was best suited for the study. Figure 4.4 is an example of how coding was applied. (A complete list of coded transcripts can be made available upon request.)


What components would you like to see go into a patch management policy?

Interviewee: I think you know a key component is what do you want to patch I think you know uhm if for instance if you run an exchange server you need to decide uhm and it needs to be granular as granular as possible so you need to decide you know I'm only gonna patch windows so I'm gonna treat applications as a separate uh s you don't have that all-encompassing patch strategy that says you know this covers all technology and all applications that I run because applications are very different to the base OS' that you run so I think key components that needs to feed into your strategy is: what do you want to address, what is the risk that you, that's why we decided on security patches critical patches kind of thing because we didn't want to you know patch uhm the windows installer and patch the windows firewall service and patch you know because those aren't even functionalities that we use kind of thing so we didn't wanna go in with a strategy and like I said any change in to your environment has a sense of risk attached to it kind of thing so we wanted to say industry best practice what should we focus on and I think that's key, is to say what is the scope of my patching gonna involve and have that for your

Figure 4.4: Example of the coding process used

4.10 Themes

The analysis of the data followed a layered approach. The first layer of analysis involved reviewing official company documents, such as the information security policy, patch management policy and patch management framework. The second layer of analysis involved qualitative research methods such as interviews and literature surveys, in order to further gain insight into the research subject. The third layer involved identifying common themes within the data.

Several themes were identified by applying latent coding. The literature was analysed by means of latent coding, and various themes were identified. The interview transcripts were analysed using the same technique, resulting in further themes being identified. The unstructured themes can be found in Appendix C and the structured themes can be found in Appendix D.

4.11 Categories

Various pieces of data were grouped together to reduce the number of data pieces that required processing in the data analysis process. Similar themes were concatenated and a record was kept of how many interviewees expressed a certain view. The number of interviewees expressing a certain view was used as a weighting (see table 4.4). Value was assigned to themes based on this weighting.

4.12 Themes emerging from the literature

After the literature was reviewed, the researcher applied intensive reading and coded several key phrases. Similar phrases were identified and concatenated into single phrases, while duplicated phrases were removed. The results of this process are presented in table 4.3, together with supporting authors. (Note that the themes in table 4.3 are not quotes, but rather themes that emerged from the literature).

Table 4.3: Emerging themes from the literature with supporting authors

Key Phrases/Themes (not quotes) from literature	Supporting Authors
Patch management will add some form of complexity to the environment.	• Chen et al. (2010) • Sommerville (2011) • Highsmith (2013)
The patch management policy should play a critical role in the overall security of an organisation.	• Gerace and Cavusoglu (2009) • Arora et al. (2010) • Felmetsger et al. (2010) • Souppaya and Scarfone (2013)
Infrastructural limitations such as available bandwidth should be considered when designing a patch management policy.	• Souppaya and Scarfone (2013) • Jenkins et al. (2014)
All risks related to patch management should be identified and classified.	• Schryen (2011) • Foley (2009)
The user community often does not apply patches for a variety of reasons.	• Bilge and Dumitras (2012) • Sethanandha (2011)
High availability systems pose a challenge to patch as they cannot be rebooted or shutdown.	• Jenkins et al. (2014) • Giuffrida et al. (2013)
Patch management is linked to enterprise data security.	• Cavusoglu et al. (2008) • Gerace and Cavusoglu (2009) • Sharma et al. (2011)
Vendors typically do not support older versions of software.	• Sommerville (2011) • Tutt (2014)
Testing plays a crucial function with patch management.	• Arena et al. (2011) • Chen et al. (2010) • Souppaya and Scarfone (2013)
Organisations have constrained resources such as funds and personnel availability for patch activities.	• Souppaya and Scarfone (2013) • Jenkins et al. (2014) • Sethanandha (2011)
Different software platforms require different approaches toward patch management.	• Thakare and Gore (2014) • Farwick et al. (2011)
There is no centralised source of patch information.	• Joshi (2013) • Le Goues et al. (2013)

4.13 Key findings from interview data

The data analysis process started after interviews were transcribed. The first step was to highlight all phrases and statements that the researcher deemed important. The data was then grouped into a table by interviewee and by question. The responses from the various interviewees were grouped together per question. Key phrases per interviewee and per question were copied in separate documents. Each statement was tagged with an interview identifier in order to maintain an audit trail of which interviewee said what.

Duplication from the same interviewee was removed, and the raw data was grouped together per question, maintaining the interviewee tag with each block of data. The raw data was then interrogated to further remove duplication or any statements that may not be in context or of value. A table was created to present the raw data statements per questions. This table was the primary input for the final data analysis.

The raw statements per question were processed with Word to identify the highest frequency words. These words were used as a guideline to interrogate the raw data table and identify common themes with the weighting attached to each theme. The weighting is a numeric value consisting of the number of participants that had the same view or opinions, and can be reverse looked up back to the interviewee for audit purposes. Quotes and exceptions from specific interviews have also been identified. Detailed proof of each step is available upon request. Figure 4.5 depicts the data analysis process that was followed.

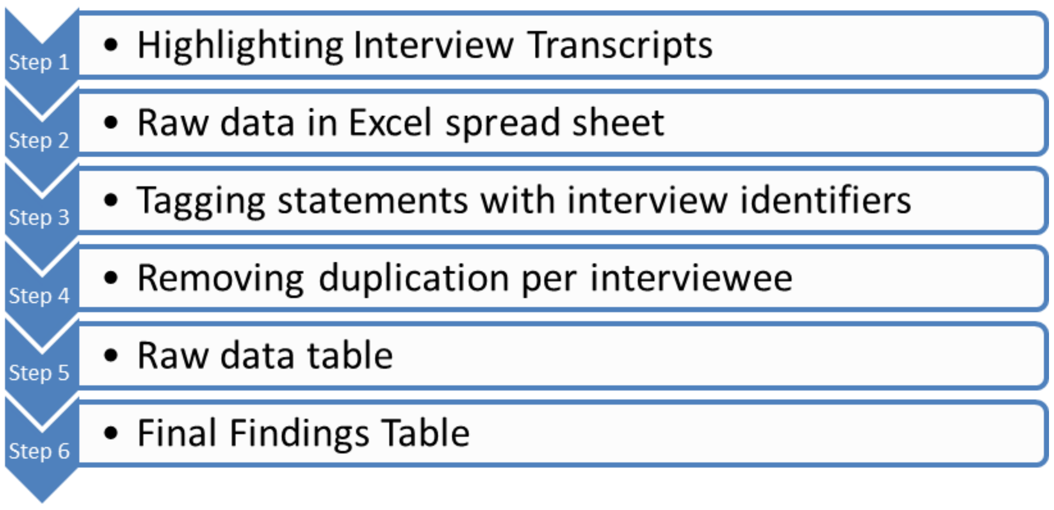

Step 1
• Highlighting Interview Transcripts
Step 2
• Raw data in Excel spread sheet
Step 3
• Tagging statements with interview identifiers
Step 4
• Removing duplication per interviewee
Step 5
• Raw data table
Step 6
• Final Findings Table

Figure 4.5: Summary of data analysis process

4.14 Final data analysis

Upon completion of the aforementioned process, the result was an Excel spreadsheet with the following headings: interview question number, common themes/categories, and the number of interviewees that were explicit and implicit. Exceptions to the identified themes were also highlighted. These tables are presented below.

The *question* refers to the interview question, and the *themes/categories* are the emerging statements. The *explicit* field represents the number of interviewees who explicitly had a specific view. The *implied* field represents the number of interviewees who implied a specific view. The *exceptions* view is the number of interviewees who had an opposite view compared to the general view shared by a large number of interviewees.

The results are represented in tables 4.4, 4.5 and 4.6 respectively, and data is sorted to display the findings in the order of strongest to weakest, determined by the number of interviewees who agreed with a specific finding. The statement number is used to link topics that are discussed in Chapter Five with specific findings in tables 4.4, 4.5 and 4.6 respectively.

Table 4.4: Explicit findings (summarised)

Statement Number	Interview Question Number	Findings	Number of interviewees who were explicit
4.4.1	10	Patches are generally downloaded from vendor's website, and in some cases with critical systems the vendor might come on site to do the patching themselves.	13
4.4.2	13	The consensus was that a single department should not develop the patch management policy; instead, it should be a combined effort and all role players should be consulted.	11
4.4.3	5	Most interviewees had an idea of the role players who were at that stage involved in the patch management process, in relation to their department. All interviewees were aware of what was being done, but did not necessary know who was doing what.	10
4.4.4	8	The testing process is not formally defined, but the informal process is to test patches in a lab environment; then do a small pilot release, followed by a gradual release into the production environment.	10

Statement Number	Interview Question Number	Findings	Number of interviewees who were explicit
4.4.5	9	The company does not have a defined patch prioritising mechanism.	10
4.4.6	4	Interviewees indicated they are very familiar with generic patch management issues within the organisation, but noted that their familiarity is largely limited to their specific domain.	9
4.4.7	5	Key roles payers (according to interviewees) were desktop support, server support, information security and store support.	9
4.4.8	7	The consensus was that Microsoft is seamless in terms of patch management, however, not all vendors are this structured in terms of releasing software patches.	9
4.4.9	2	Several interviewees indicated they were very familiar with generic patch management issues and could elaborate with examples.	8
4.4.10	12	The patch management policy should define the roles and responsibilities clearly.	8
4.4.11	13	Information security should be accountable at a stakeholder level, while the technical implementation of patch management tasks should be completed by the operational teams.	8
4.4.12	11	In the past a worm (Conficker) broke out and caused havoc on the network that could have been avoided if the company had an active patch management strategy in place.	7

Table 4.5: Implied findings (summarised)

Statement Number	Interview Question Number	Findings	Number of interviewees who implied a view
4.5.1	11	There is risk associated with having an active patching strategy as well as risk associated in not having an active patching strategy.	11
4.5.2	3	Patch management activities are inconsistent, and the time being spent on patch related activities varies from person to person.	9
4.5.3	4	It is often challenging to get patch deployments approved within the company because patches have broken many working systems in the past.	7
4.5.4	2	Patch management is generally complicated and often comes with a variety of challenges.	4

Table 4.6: Exceptions to the implicit and implied findings (summarised)

Statement Number	Interview Question Number	Findings	Interviewee Identifier
4.6.1	9	The company has a prioritising mechanism in place.	3
4.6.2	6	The Information Security department should be involved.	2
4.6.3	5	One interviewee was not aware of who the role players were.	1
4.6.4	2	There are no patch management issues within the company.	1 (i6)
4.6.5	3	The company has an active patch management strategy.	1 (i4)
4.6.6	8	The company have a defined testing strategy.	1 (i12)

4.15 Mapping the findings to themes

For the convenience of the reader, table 4.7 presents a mapping of the findings and the corresponding statement number to the themes. The themes will be discussed in greater detail in Chapter Five.

If a finding was “the consensus was that a single department should not develop the patch management policy, instead it should be a combined effort and all role players should be consulted”, the theme deducted from this finding would be “patch management policy development stakeholders”. As another example, if a finding was ”the testing process is not formally defined, but the informal process is to test patches in a lab environment, then to do a small pilot release that is followed by a gradual release into the production environment”, the themes deducted from this finding was “the importance and challenges of testing patches”.

For a comprehensive mapping of the findings to the themes, see table 4.7.

Table 4.7: Mapping findings to corresponding themes

Statement Number (from table 4.4, 4.5 and 4.6)	Findings	Corresponding Theme
4.4.1	Patches are generally downloaded from a vendor’s website, and in some cases with critical systems the vendor might come on site to do the patching themselves.	Sources of software patches and related information

Statement Number (from table 4.4, 4.5 and 4.6)	**Findings**	**Corresponding Theme**
4.4.2	The consensus was that a single department should not develop the patch management policy, instead it should be a combined effort and all role players should be consulted.	Patch management policy development stakeholders
4.4.3; 4.4.6; 4.4.10; 4.4.11; 4.6.2 and 4.6.3	Most interviewees had an idea of the role players who were at that stage involved in the patch management process in relation to their department. All interviewees were aware of what was being done, but did not necessary know who was doing what. Interviewees indicated they are very familiar with generic patch management issues within the organisation, but noted that their familiarity is largely limited to their specific domain. The patch management policy should define the roles and responsibilities clearly. Information security should be accountable at a stakeholder level, while the technical implementation of patch management tasks should be completed by the operational teams. The Information Security department should be involved.	Roles and responsibilities
4.4.4 and 4.6.6	The testing process is not formally defined, but the informal process is to test patches in a laboratory environment; then do a small pilot release, followed by a gradual release into the production environment. The company has a defined testing strategy.	The importance and challenges of testing patches
4.4.5 and 4.6.1	The company does not have a defined patch prioritising mechanism. The company has a prioritising mechanism in place.	Patch deployment prioritising mechanism
4.4.6; 4.4.9; 4.6.4 and 4.6.5	Interviewees indicated they are very familiar with generic patch management issues within the organisation, but noted that their familiarity is largely limited to their specific domain. Several interviewees indicated they were very familiar with generic patch management issues and could elaborate with examples. There are no patch management issues within the company. The company has an active patch management strategy.	Interviewee familiarity: patch management issues
4.4.8	The consensus was that Microsoft is seamless in terms of patch management; however, not all vendors are this structured in terms of releasing software patches.	Software vendor behaviour and their Effect on patch management
4.4.12 and 4.5.4	In the past a worm (Conficker) broke out and caused havoc on the network that could have been avoided if the company had an active patch management strategy in place. Patch management is generally complicated and often comes with a variety of challenges.	Patch management challenges
4.5.1	There is risk associated with having an active patching strategy as well as risk associated in not having an active patching strategy.	Risks related to patch management

Statement Number (from table 4.4, 4.5 and 4.6)	**Findings**	**Corresponding Theme**
4.5.2	Patch management activities are inconsistent, and the time being spent on patch related activities varies from person to person.	Consistency of patch management activities
4.5.3	It is often challenging to get patch deployments approved within the company because patches have broken many working systems in the past.	Importance of patch management

4.16 Chapter summary

This chapter discussed the data collection process that was applied to the literature review, interview transcripts and organisational documents. Insight was given into the case that was selected, as well as the sampling method that was adopted to select the case and participants. The interview method was used to collect data from participants, and latent coding was applied to interview transcripts in order to identify findings.

This chapter described in detail how findings were identified from the interview transcripts and organisational documents. Strong ideas from the literature together with supporting authors were also presented in table 4.3. It became evident that all role players were not fully aware of the patch policy and process within the organisation. Testing was noted as an area of concern and the consensus was that the patch management policy should be developed with all role players being consulted and not in isolation by a single department.

Interviewees implied that there is risk in both patching and not patching, and that these should be weighed against each other. Some interviewees had some opposing views, indicating a possible misalignment. The final findings are presented in tables 4.4, 4.5 and 4.6, and these findings and matching statement numbers are mapped to corresponding themes in table 4.7. The themes form the basis of the discussion in Chapter Five.

CHAPTER FIVE: DISCUSSION

5.1 Introduction

For the convenience of the reader, the research questions and a summary of the research problem statement are repeated below:

Research problem statement

Organisations often find themselves running IT systems that may either be unstable or prone to intrusion because of challenges and complexities involved in patch management at an enterprise level.

Research questions

1. What causes the management of software patches to be complex?
2. How do organisations implement patch management in order to enhance enterprise data security?

Based on the data analysis process described in the previous chapter, the final data analysis findings were presented. The final analysis was grouped in three sections: explicit findings, implied findings, and exceptions. These results are presented in tables 4.4, 4.5 and 4.6 respectively. Table 4.7 presents a mapping of the findings and the corresponding themes. All of the aforementioned tables include an identifier called the *Statement Number*, and the identifier is indicated next to the themes below between brackets, in order to make the link between Chapter Four and Chapter Five explicit. The chapter considers the following themes:

i. Sources of software patches and related information (4.4.1)
ii. Patch management policy development stakeholder (4.4.2)
iii. Roles and responsibilities (4.4.3; 4.4.6; 4.4.10; 4.4.11; 4.6.2 & 4.6.3)
iv. The importance and challenges of testing patches (4.4.4 & 4.6.6)
v. Patch deployment prioritising mechanism (4.4.5 & 4.6.1)
vi. Interviewee familiarity: patch management issues (4.4.6; 4.4.9; 4.6.4 & 4.6.5)
vii. Software vendor behaviour and their effect on patch management (4.4.8)
viii. Patch management challenges (4.4.12 & 4.5.4)
ix. Risks related to patch management (4.5.1)
x. Consistency of patch management activities (4.5.2)
xi. Importance of patch management (4.5.3)

5.2 Themes developed

5.2.1 Sources of software patches and related information

Current software inventory information was noted as a required input to an effective patch management system (Souppaya & Scarfone, 2013). Inventory reports are often skewed for reasons such as the fact that most configuration management tools rely on a software client to be installed on the client machines. If the software client is broken, no client machine information can be obtained (Huang et al., 2012). I13 confirms that this is a problem within the company and said that "the problem is actually maintaining a software asset register and then following up with each and every one of those application vendors to see if they've got patches" (see Appendix I).

Joshi (2013) indicates that various sources provide software patches and related information about available patches and software vulnerabilities. These repositories include government websites, technical blogs, security bulletins and software vendor websites. Although it is common practice for vendors to publish patch-related information and patch releases on their websites, some third-party sites such as the National Institute of Standards and Technology (NIST) and Common Vulnerabilities and Exposures (CVE) have attempted to collate this information into central repositories. This is considered unofficial and not all vendors are catered for. Joshi (2013) further states that in order for administrators to have complete information about software patches and system vulnerabilities, administrators have to monitor numerous sources of information. I9 supports this claim and referred to software vendor websites by saying that "we're also signing up to their patch management notifications" (see Appendix I).

Keeping current with patches can be a challenging task. It is therefore important to have current information that will allow for quick decision-making with regard to patch management. I9 supports this statement and said that Microsoft products allows for automation to an extent by saying that "our WSUS server which is attached to the SCCM server will then get that content for us and download it, and based on that we'll create templates" (see Appendix I).

5.2.2 Patch management policy development stakeholders

The first step towards developing a patch management strategy is typically to define a patch management policy within an organisation. Souppaya and Scarfone (2013) state that for a patch management policy to be effective, it must be endorsed by senior management within an organisation. Furthermore, the policy should provide guidelines for the role and responsibilities, the testing of software patches, and the patch deployment procedures. When considering who is responsible to develop the patch management policy, I1 asserted that ”it is a combined effort with the operations teams involved so and the server team obviously, there’s the desktop support team and, uhm, the security department as well so those are the people that need to basically define that policy” (see Appendix I).

The majority (11 out of 13) interviewees agree that that the patch management policy should not be developed by a single department. Instead, it should be a combined effort with all role players consulted in the process. I3 stated:

> “I think security should develop the policy but all parties involved must be consulted for input” (see Appendix I). Some interviewees felt that if the policy is developed in an isolated manner, it may be unrealistic and ignored by the other role players all together. I5 suggested that the information security department be consulted, but they should not own the process of developing a patch management policy. I5 further stated that “an independent department should draw up the policy and all the role players should be consulted” (see Appendix I).

Operational teams have insight into the current challenges in terms of patch management within the company, and are generally responsible for the technical implementation of the policy. I4 supports the idea of involving operations teams in the policy development process, stating that “I think we should use the practical experience of operations teams” (see Appendix I). As a result, the operational team is able to make a valuable contribution to the development of the patch management policy. Patch management may be more important in some organisations than what it is in others. For this reasons the scope and importance of patch management should be determined within a specific organisation by means of a patch management policy. I3 confirmed that this is applicable to the company and said that “the importance of patching is not always well understood” (see Appendix I). It may be helpful to determine a set of basic principles that is organisation-specific. I3 said that:

> “…the policy should reflect the importance of patching, people don’t understand it and therefore they simply ignore it so the policy should say why we need to patch, the policy should define roles and responsibilities and furthermore it should state the frequency of patch deployments; it should also state the scope of what we need to patch, and we may choose not to patch everything” (see Appendix I).

Souppaya and Scarfone (2013) assert that a policy should state who is responsible for monitoring and testing as well as patch deployment alternatives. Effective monitoring could allow for patches to be tested as soon as patches become available to the public. I5 confirmed this statement and said that “the policy should also define the technique strategy and emphasise the importance thereof and then the roles and responsibilities” (see Appendix I).

Although the company does have a patch management policy in place, the lack of a policy was noted as a concern by a few interviewees. This could be interpreted as a lack of awareness of the patch management policy and/or the contents thereof. The organisation had both an information technology security policy and a patch management policy, but not all of the role players were aware of the patch management policy. I7 said that “I know security is busy drafting a policy but the whole process is still at its early stages” (see Appendix I). I3 said: “I haven’t seen the policy but it is somewhere on the intranet I believe” (see Appendix I), while I8 made a similar comment: “I haven’t actually seen the patch management policy” (see Appendix I). A possible approach might be to have one department assume the responsibility of drafting the policy, and then consulting all the stakeholders in the process.

5.2.3 Roles and responsibilities

Souppaya and Scarfone (2013) indicate that roles and responsibilities should be as detailed as possible to avoid confusion. Sihvonen and Jäntti (2010) support the notion that the roles and responsibilities should be defined in words and in as much detail as possible. Several interviewees (8 out of 13) felt that the patch management policy should explicitly define the roles and responsibilities in a clear and concise manner. I3 said that “I think names should be associated to certain roles” (see Appendix I). I1 further stated that the “only concern is roles and responsibilities” (see Appendix I). I7 said: “I do feel like roles and responsibilities should be clearly defined” (see Appendix I). When considering who should be added to the patch management process, I13 indicated that “I’ll only add people from an information awareness point of view” (see Appendix I).

A large number (10 out of 13) of interviewees are aware of who the role players are within the company that is currently involved in the patch management process in relation to their department, while a few interviewees (3 out of 13) are not as informed as the rest in terms of the patch management role players. I8 said that, “I think security ought to be involved more than they are, uhm, even if it’s just not necessarily from a doing point of view but from a motivating point of view" (see Appendix I). All interviewees agreed that the current role players in the patch management process are desktop support, server support, information security, and store support. Two interviewees felt that application developers should be part of the process as well. Although this is compulsory from an in-house application point of view, it may not add much value form an external vendor application perspective.

I3 said that "I think everyone is already involved that that should be, uhm, but I didn't or I don't think it's defined properly; I think names should be associated to certain roles and people should be held accountable for their responsibility" (see Appendix I). Information security is involved in the patch management process. This indicates an awareness issue among interviewees in terms of the patch management activities. A patch management policy could thus bridge the awareness gap. The danger of roles and responsibilities not being clearly defined is that stakeholders may not be aware of who is doing what, and this could lead to a duplication of work or result in the omission of work, as one team could wrongfully assume that another team is doing the work.

After analysing the data, it became evident that no central department is looking at the overall bigger picture. Focus is specific to the domain and task at hand, and all activities were performed in these isolated environments. A RACI matrix could be used to simplify the process of defining roles and responsibilities. I1 supports this statement by saying that "in that RACI model there's usually a table, but define in words who's responsible for what from a policy point of view" (see Appendix I).

5.2.4 The importance and challenges of testing

A large number of interviewees (10 out of the 13 interviewees) mentioned testing as an area of concern. The testing process is not formally defined, but the informal process is to test patches in a laboratory environment, followed by a pilot and then a gradual release to the production environment. I1 elaborated on this:

> "…first patch is just to test the process and make sure the application is installed, not the application, the patch is installed and that there aren't any problems in the deployment process once that's done we move on to pilot 2 if you want to call it that or phase 2, uh, where we choose specific users in our business; preferably people that use a large variety of applications in that business unit" (see Appendix I).

Souppaya and Scarfone (2013) assert that testing is a time-consuming task and places a degree of strain on a company's resources. The danger exists that a manager might not deem the testing as important as other operational tasks, resulting in patches being released that were not properly tested. I10 said that "we've had a couple of incidents where we haven't tested things properly so we've actually applied a patch and there might have been an issue as the result of a patch and that's typically because we didn't test it properly" (see Appendix I). I7 reiterated the importance of testing by explaining an example where the company "deployed a bug fix for IE8. This was still on XP and the install prompted for a reboot and all the machines gave a blue screen error; fortunately we created regular restore points so we could roll back easily" (see Appendix I).

Le Goues et al. (2013) note that the test environment often does accurately reflect the production environment. I5 asserted that the test environment is a challenge, and mentioned an example where a patch failed that was tested. If the test environment is not aligned to the production environment, the behaviour of patched systems may differ in test and production environments. The result is that even if a patch is extensively tested in the test environment, it can have a different effect in production. I5 said that "we do have tests; yes the problem is we can test an installation on these test boxes but we don't have users testing the apps on the boxes, so the true test is only really when we deploy to production" (see Appendix I). I5 gave an example where a patch was tested, but still caused problems in the production environment after it was deployed:

> "We deployed a fix to the exchange server and suddenly all outlook desktop lines started prompting for new user names and passwords; turns out this patch actually broke the Kerberos protocol, I know it's important to know we did test this patch on our server and passed our test; it installed successfully but that's the thing about applications - they mustn't just install they must actually work" (see Appendix I).

This highlights the importance of doing extensive testing, and not just basic deployment testing. Often various use cases have to be tested to make sure all the functionality continue to work after a patch has been deployed.

From the vendor's point of view, limited testing can be done in the sense that a vendor can only test functional components within a system. Oracle, for example, can test that the Oracle functionality works, but the vendor cannot with certainty say how a software patch will behave in a client's environment. Software and hardware platforms as well as current systems and patches installed all play a role in this regard. For this reason testing remains an important function on the client side.

5.2.5 Patch deployment prioritising mechanism

Prioritising which patch to install and when to install the patch is closely related to each other. The criticality of the affected system should be considered, as well as the severity of the vulnerability that the system is exposed to. Souppaya and Scarfone (2013) assert that dependencies of the patches should be considered as it may be that the installation of one patch requires another patch to be installed first, or that the system needs to be rebooted pre- or post a patch deployment.

Not all interviewees appeared to know whether the company had a patch prioritising mechanism in place. I3 said that “we don’t have a formal mechanism for prioritising patches; we only do security and critical patches from Microsoft; if it’s a patch that fixes an active issue in our environment then we deploy as soon as possible; normally within a week” (see Appendix I). I4 confirmed this:

> “…we don’t prioritise where we say every six months we patch with the latest critical and, uhm, you know security patches that’s our policy so we don’t prioritise in that, however if we get for instance we hit a snag or an issue, uhm, be it anti or virus related, malware related, uhm, you know because sometimes there’s windows security patches also for a vulnerability that gets highlighted then typically our governance, our xxxxxxx’s *[name omitted]* team, our security and governance team, they’re responsible and they will bring it under our attention” (see Appendix I).

The company’s patch management policy does address prioritising patches. This indicates that not all interviewees are familiar with the content of the patch management policy. Furthermore, I8 raised a concern that “the business changes always trump the patching, so if we say we’re gonna do patching on this day, it’s fine until there’s business requirements” (see Appendix I). Patch management policy could provide guidance in terms of prioritising patches, and further help to protect timeslots and other resources that are allocated to patch management activities.

Adobe has a rating system called the Adobe Rating System. According to Schryen (2011), this system serves as a guideline for customers with managed environments. The priority ranking is based on historical attack patterns, the type of vulnerability, and the platform and potential mitigations that may be in place. According to Adobe, a Priority 1 is defined as an update that resolves a vulnerability being targeted, or that may have a higher risk of being targeted. Adobe recommends that Priority 1 patches be installed within 72 hours of its release. A Priority 2 is defined as an update that resolves vulnerabilities that have been at elevated risk in the past. Their recommend time for patching this vulnerability is 30 days. A Priority 3 patch is defined as an update that fixes vulnerability that has not been a target for attack in the past. I5 said that “we only do Microsoft patches and we only do security and critical patches; obviously if a patch is fixing something that is broken then you need to apply it as soon as possible” (see Appendix I).

Similarly to Adobe, Microsoft has its version of a prioritising guideline, and Google might have theirs. The challenge is for organisations to develop their internal patch prioritising strategy, whilst aligning with all their software vendors.

5.2.6 Interviewee familiarity: patch management issues

A large number of interviewees (9 out of 13) indicated they are very familiar with patch management issues specific to the organisation, but noted that their familiarity is largely limited to their specific domain. Souppaya and Scarfone (2013) state that some companies still do patch installation manually, and although there are benefits to this approach, at some point in time the cost could outweigh the benefits. I6 said that “if you’re gonna install automatically, install stuff like service packs and drivers and all other kinds of funny stuff, er, that you haven’t properly tested then you’re gonna get some issues” (see Appendix I).

Several interviewees (8 out of 13) indicate they are very familiar with generic patch management issues and could elaborate with examples. The technical knowledge was evident; people know how to do what needs to be done although the central view was not being observed by anyone.

I1 shared some insight into the patch management dependencies in the company:

> “…patch management obviously adds a complexity to that environment; we need to understand that if something goes wrong it’s not only my team that’s gonna have to fix it; either we have to roll back but at the same time we need to engage with development teams and say okay this patch was applied, this is what it’s done, you need to go back and re-develop or re-engineer your application so that it doesn’t use that loophole or vulnerability” (see Appendix I).

5.2.7 Software vendor behaviour and their effect on patch management

All interviewees agree that patches are downloaded from the vendor’s website, and in a few instances the vendor would come on-site to apply the software patches themselves. The vendors would typically come on-site for mission critical systems such as the Oracle database or the store area network. I10 said that “some of those mature vendors have, uhm, processes in place where they actively pick up issues and they release patches" (see Appendix I).

Microsoft is seamless in terms of patch management, however, not all vendors are this structured in terms of releasing software patches (9 out of 13 interviewees). Microsoft releases patches every second Tuesday of the month, while some other software vendors who do not have defined patch release dates, release software patches in an ad-hoc manner (Zseby et al., 2013). This adds a level of challenge for organisations as they almost always have to be ready for emergency patch releases. I4 confirms this statement by saying that as an organisation, you typically "have to align with the application vendor of the application" (see Appendix I).

Microsoft has a solution called WSS. It is a Microsoft-only solution and only supports Microsoft applications. I13 explained that "W-SASS only works with Microsoft so you can't actually deploy like Adobe or semantic patches through W-SASS, so it's only for Microsoft "(see Appendix I). The view is that different vendors have inconsistent behaviour in terms of patch management. I11 said that "Microsoft patch is they come down and they then you get things like Adobe which just spring in to life and just patch every now and again" (See Appendix I).

The challenge for the organisation is that the organisation is forced to align with the vendor's patch release behaviour. It is also common for vendors not to release patches for old versions of software. According to Tutt (2014), the most notable example of this situation is Windows XP. Microsoft ended the support for Windows XP in April 2014 and will not release any further patches for this operating system.

5.2.8 Patch management challenges

Due to the impact of patch management challenges on the entire patch management process, this section will consider various patch management changes in greater detail in sub-sections.

The high rate of vulnerability announcements has placed the focus on matters such as vulnerability disclosure, the speed of patch generation, and the dissemination of patches (McQueen et al., 2009). When considering patch management challenges, the scope of the challenges increases as the number of different software platforms increase within an environment.

I4 confirmed this statement:

> "It's easier if you run typical Microsoft workload for instance, so if you run an exchange server or a domain controller, activating domain controller, to patch them consistently, that's typically not an issue because its Microsoft with Patches for Microsoft, but where you get challenges is when you running third party applications" (see Appendix I).

Insufficient software inventory information presents a challenge because patch management is dependent on current inventory information with a given environment (Souppaya & Scarfone, 2013). The below section will discuss various patch management challenges identified in this study in more detail.

5.2.8.1 Resources, infrastructure and tools

Souppaya and Scarfone (2013) indicate that organisations often have limited resources and lack the understanding of how important patch management is. With limited resources and infrastructure, patch deployments are often in competition with other tasks such as software version upgrades and antivirus definition updates. Limited resources place a fair amount of strain on IT managers when deciding what projects to run and which to cut. I1 said that "Microsoft recommends that you patch every Tuesday; that's not practical in a corporate such as ours, you'd probably need a dedicated patch management team and you know here we sort of use our resources for everything possible" (see Appendix I).

Microsoft System Centre Configuration Manager (SCCM) is an example of a configuration management tool that allows for Microsoft software patches and applications to be packaged and deployed. Configuration management tools provide functionality to automate certain components of the patch management process. The cost of the tools has a direct impact on the cost of the patch management process as a whole. In addition to the cost of the tools, skilled employees are required as well to manage these tools (Sihvonen & Jäntti, 2010).

I3 said that "our biggest challenge, sorry, is that we are geographically scattered and we don't have the infrastructure that we would wish to have" (see Appendix I). This means that even if the company wanted to deploy patches in accordance with the recommendation of Microsoft and or other vendors, it simply may not be possible due to current infrastructure limitations. I13 defended this statement saying "it's a very well mitigated risk in our environment" (see Appendix I), while making reference to compensating controls such as Intrusion Detection Systems (IDS) and perimeter firewalls.

5.2.8.2 No all-inclusive off-the-shelf solution

Hosek and Cadar (2013) assert that there is no one-size-fits-all solution to address all the challenges of patch management. I13 said that "when Conficker came out but, uhm, we actually had issues where machines weren't patched so, uhm, and then the antivirus on the machines weren't up to date where Conficker brought down our environment and half of our store's environment for more than two days" (see Appendix I).

One of the challenges with vendor specific tools is that it provides solutions only for their products. If the number of third party software vendors increase within an organisation, vendor specific solutions may not be the best option. Considering the large number of applications in the environment by different vendors, and the geographical disparity of client within the environment, patch management activities within the company would just be ceased all together. I3 confirmed this by saying: "I know for a fact with XP we never patch, we did years ago then we fell behind and then we just stopped, uh, we are hoping now to change this with Windows 7 deployment" (see Appendix I).

5.2.8.3 Communication

The patch management process involves various stakeholders from different IT departments, business units and software vendors. For this reasons a well-defined communication strategy is important (Sihvonen and Jäntti, 2010). I9 said that "we don't have that 360 degree feedback" (see Appendix I).

A few interviewees (3 out of 13) felt that the Security Department was not as involved in the patch management process as they ought to be. According to the Security Department, however, they were actively driving the strategy and were well-informed of all challenges within the organisation. While both operational teams and the security department appeared honest in their response, the problem became evident. There was a lack of communication and awareness – no-one was looking at the bigger picture. When asked if people should be added to the patch management process, I13 said that "I'll only add people from an information awareness point of view but I think we first need to get the process established and off the ground before we go to that, uhm, level" (See Appendix I).

I3 gave an example of where the testing strategy failed in the past:

> "Recently in the beginning of the year we deployed a JAVA patch tested on the virtual machine; all seemed fine, deployed to a pilot of five, no communication made it back to us, interesting thing happened here - three of these five people logged a call because on a web based application because a web based application no longer worked, desktop support did a system restore, removed the patch and the issue was resolved; all this time we were under the illusion that we have a hundred percent success rate so far, the next week we deployed to thirty people, uhm, then it became an issue because thirty users could now not use the realist system, so here although we did test, our communication strategy failed us" (See Appendix I).

All stakeholders appeared to be doing what they were supposed to do, but the communication mechanism is assumed rather than defined. There is thus no way of tracking communication, as it is often just an email from one party to another. This is an area of concern because a situation could arise when important communication does not make it to the intended party, resulting in certain functions not being completed.

5.2.8.4 Maintenance timeslots

It is important for maintenance windows to be identified, discussed and agreed upon upfront with business stakeholders. Maintenance windows are noted as a concern by a few interviewees. I8 said that "our biggest challenge in the stores environment is getting the time slots to do it because there's so much change being driven in the environment by the business". I8 further said that "security ought to be involved to protect the patching from being superseded by a business requirement" (see Appendix I).

Shahriar and Zulkernine (2012) highlight the importance of finding the right timeslot in which to do patch deployments. A few interviewees reported that an area of concern is to acquire timeslots in which to do patch deployments. I7 said that "one of our biggest challenges is time slots; at the moment business needs always to precede patch deployments, our second challenge is bandwidth; we simply don't have the infrastructure to push patches to every single store" (see Appendix I). Maintenance windows are typically scheduled at times least disruptive to the business and include enough time for the task, as well as rollback procedures.

If the patch management policy is endorsed by top management, it could reduce resistance and result in patch deployments being approved more easily (Souppaya

& Scarfone, 2013). I5 supported the view that if the patch management policy is endorsed by senior management, it may reduce resistance towards patch management. I5 said that "by showing the commitment of senior management people are more likely not to push back on proposed patch deployment" (see Appendix I).

5.2.8.5 Legacy systems

Legacy systems are old systems that are still in use and often provide business critical functions; therefore they cannot easily be decommissioned or replaced. Vendors generally do not support these systems anymore, leaving them vulnerable to exploitation (Sommerville, 2011).

Legacy systems pose great risks and challenges to organisations in terms of patches, because vendors often do not support these systems (Souppaya & Scarfone, 2013). I4 said that "there's definite risk also in an active patching strategy because you've got Legacy applications" (see Appendix I).

A good example of a problem with legacy systems is the fact that Microsoft ended the support for Windows XP in 2014. Windows XP is one of the most popular operating systems and is still used (at the time of this writing) even after Microsoft has stopped supporting it, making it a vulnerable operating system. Windows XP can be considered a legacy system (Tutt, 2014).

I1 said that patch management is "a complicated thing to actually manage especially in a big corporate such as ours at the moment, uhm, unfortunately we've got a lot of disparate applications, some of them Legacy" (see Appendix I). Legacy system challenges can be mitigated to an extent. If systems cannot be updated for whatever reason, compensating controls can be put into place.

5.2.8.6 The people aspect

One of the findings was that not all patch management challenges were technology-related. Some challenges turned out to be people problems, more specifically, a problem with the mentality of certain role players. The main reason for these types of challenges is often related to preconceived conceptions that are formed as a result of past experiences. I1 gave an example of a past experience and said that "I've experienced it where we patched something; it's broken IE to hell and gone" (See Appendix I).

I5 stated that:

> "…the biggest challenge for me is getting the change approved in CAB, they are very reluctant to approve because generally they perceive that patches don't add any value although they have the ability of breaking something now; with that being said it's very difficult to justify the deployment of patches in production servers" (see Appendix I).

It is important for the non-technical challenges to be considered and addressed as part of the overall patch management strategy.

> "…we have in the past been a bit suckered where we haven't applied patches properly and then when you do come to the point where you have to apply the patches you're so far behind that it becomes, it doesn't become worth patching then it's a big and expensive exercise of re-installing operating system" (I11, see Appendix I).

I8 has the view that more people in the organisation should understand the importance of patch management and said that "more parties need to understand patching, uh, they need to understand the implication of patching or the implication of not patching" (see Appendix I). I10 supports this notion and said that "one of our issues is that we don't patch enough" (see Appendix I).

I4 also had a similar experience where applying patches caused disruption and said that "we've rather had issues patching servers, to be quite honest with you, than not patching servers so the counter is actually more true in my practical experience" (see Appendix I). As a result of negative past experiences, the overall approach towards patch management is cautious, and business stakeholders tend to exercise more resistance towards proposed patch deployments.

5.2.9 Risks related to patch management

There is risk associated with having an active patching strategy as well as risk associated with not having an active patching strategy (11 out of 13 interviewees). I8 said that "we're at as much risk if we don't patch as if we do patch" (see Appendix I). Deploying patches hold the risk of working systems breaking, and not deploying patches hold the risk of vulnerabilities being exploited.

In some instances an organisation would decide that there would be greater risk in not deploying patches immediately than there would be in deploying without testing. This decision is risk-appetite dependant and would vary from organisation to organisation.

> "I know that we've had issues with software, whether it's platform software like Biztalk or sequel server, uhm, where we could have actually resolved those issues if we had been pro-active around applying patches. I also know that we've had issues where we've actually applied patches and had issues because of it" (I10, See Appendix I).

According to Cárdenas et al. (2011), a risk assessment should be completed on all servers on the network. This assessment should ideally include critical data stored on servers, impact of downtime that can be introduced by the server, and the general vulnerability of the server in terms of internal and external attacks. I4 mentioned an example:

> "If you haven't patched for long period like where we started a year ago, then your immediate risk is that the server has been running for seven years like this and now you patch it; there's about 85 some of our servers had about a 135 updates that needed to apply so you're touching 135 sets of DLL's and sorts of things, so chance is actually big that you might affect something on that application, but I think once you're active, once you're in your second and third cycle, it becomes a no brainer" (see Appendix I).

Foley (2009) asserts that security governance is responsible for prioritising and managing risks across an organisation. The risk of applying patches to a working system and then breaking it, versus the risk of not applying patches to a system and then leaving it vulnerable, ought to be considered.

5.2.10 Consistency of patch management activities

The consistency of patch management activities in the company was noted as a problem. I6 said that "we are not in a position where we patch very often, we haven't been in a, there've been issues in the past with applications and issues with patches and the guys just stopped patching" (see Appendix I). Beres and Griffin (2012) state that certain vendors may decide to release patches in a bundle once every cycle, where a cycle can be monthly, quarterly or even yearly. An example of this is a patch release by Oracle for the month of October 2013. This patch contained 127 new bug fixes (Oracle, 2013).

Ransbotham and Mitra (2013) state that if vulnerabilities are publicly disclosed without the vendor responding in due time, it can create a window of opportunity for malicious users. I3 confirmed this by saying, "there are times when we do patch and then for a month or so we would actively be busy and then eight months could go by and no-one talks about patching" (see Appendix I).

According to Zhu et al. (2011), the timing between the detection of a vulnerability and the availability of a patch for the vulnerability becomes crucial for the assessment of the security risk exposure of software users. Not all vendors have predefined dates for patch releases (Zseby et al., 2013). Inconsistent vendor behaviour could affect the consistency of organisations as organisations are often forced to align with vendors.

5.2.11 The importance of patch management

Although there are many reasons for patching, system stability and security are the most common reasons for applying patches (Saleem, Yu & Nuseibeh, 2012). As important as patches are, many in the user community simply do not apply patches (Bilge & Dumitras, 2012). I8 is of the opinion that a well-defined policy endorsed by senior management can help to reduce the resistance towards patch management. I8 said that “if you’ve got a policy that says well we have to do it then you can hold the policy up and say no sorry I can’t move it because the policy says I have to do it” (see Appendix I).

I12 explains that failing to maintain current patches can result in malware infections and security breaches. I12 said that “the most frequent and visible sign is a worm-breakout when a, when you have a worm breakout on the network you can easily see which systems or work stations or which servers is the most affected and those are the ones typically that’s the worst patched” (see Appendix I). I13 supports this notion by referring to another incident, and said that “when Conficker came out but, uhm, we actually had issues where machines weren’t patched so, uhm, and then the antivirus on the machines weren’t up to date where Conficker brought down our environment and half of our store’s environment for more than two days” (see Appendix I).

In the past the Conficker worm broke out and caused havoc on the network; that could have been prevented if the company had an active patch management strategy in place (7 out of 13 interviewees). This is related to patches not being actively applied, for a variety of reasons. “If we don’t suffer from a problem, we typically wouldn’t apply it” (I10, see Appendix I). It is often challenging to get patch deployments approved within the company because patches have broken many working systems in the past (7 out of 13 interviewees). I3 said that “the importance of patching is not always well understood” (see Appendix I). If patching is not well understood, functional requirements may supersede patch deployment and timeslots allocated to patching could be cancelled and used to deploy functionality instead.

5.3 Chapter summary

Patch management adds a level of complexity to the environment. The risk exists for working systems to stop working and for various unforeseen events to occur. With the high demand for availability of certain systems, it makes it more challenging to motivate for the patching of these systems. On the other hand, if the systems are not patched in a timely fashion, it could result in downtime anyway. There is thus risk associated with actively patching, as well as risk associated with not actively patching.

The three main aspects to the patch management challenges were identified as people, processes and technology. In the current patch management process within the company, there are too many decision-makers. The multiple decision-makers model introduces many undefined variables and could result in an un-patched and vulnerable environment.

Due to a lack of a defined communication strategy, certain tasks may be overlooked while others may be duplicated because of the lack of consistency in how and whether activities are completed. There are no post-deployment controls in place. The process is not explicitly owned by any department and many of the activities are performed in isolation. The result of the current process is an unstable environment, with different variations and versions of the same software running within the production environment. These vulnerable systems can be exploited at any given time, directly comprising the level of data security within the organisation.

Challenges that complicate patch management often lead to compromises in security that could easily have been prevented. If organisations can minimise the time spent on patching activities, they can utilise that time to improve other security efforts. Many organisations have largely operationalised their patch management efforts with the aim of making it a core information technology (IT) function as opposed to a function that is part of security. The following chapter addresses the research questions and sub-questions, and proposes a solution to the research problem.

CHAPTER SIX: RESEARCH QUESTIONS, RECOMMENDATIONS AND CONCLUSION

6.1 Introduction

For the convenience of the reader, the research question, sub-questions, and a summary of the research problem statement are repeated below. The answers to the research question and sub-questions form the basis of this chapter.

Research problem statement

Organisations often find themselves running IT systems that may either be unstable or prone to intrusion because of challenges and complexities involved in patch management at an enterprise level.

Table 6.1: Research question and sub-questions summary

Research Question 1	
1.	What causes the management of software patches to be complex?
Sub-Questions	
1.1	What challenges do organisations face with regard to patch management?
1.2	What challenges are created by a multi-vendor environment?
1.3	What controls can organisations implement in order to deploy the relevant patches?
Research Question 2	
2.	How do organisations implement patch management in order to enhance enterprise data security?
Sub-Questions	
2.1	How do organisations prioritise patches for deployment?
2.2	How do organisations identify patches best suited for their security needs?
2.3	Who are the key role players in managing the patch deployment process?

Based on the data collected and analysed, the previous chapter discussed the themes that emerged in further detail. This chapter starts with a discussion (section 6.2) of the patch management challenges that are relevant in terms of the research sub-questions. The reason for addressing patch management challenges is because the challenges overlap with all the research questions and sub-questions. In addition to the patch management challenges, this chapter will answer the research questions and sub-questions, make recommendations, and propose a possible solution to the research problem.

The proposed solution addresses some of the challenges and complexities of the current patch management process within the company. The chapter concludes with the author’s view on possible future research.

6.2 Patch management challenges

Due to the impact patch management challenges have on the research questions and sub-questions as well as the overlapping nature of patch management challenges in terms of single and multi-vendor software environments, the author found it appropriate to present the patch management challenges before answering the research sub-questions,

6.2.1 Resources, infrastructure and tools

It emerged that both resources and the current infrastructure within an organisation can cause constraints in the patch management process. Limited resources can be in the form of human resources, monetary resources or infrastructural resources such as available bandwidth. Organisations often have to provide highly available and secure services to business units while being constrained by the available budget and current available infrastructure. The challenge for IT managers is to decide what IT projects to run, while considering the available resources and still maintaining a secure and stable environment.

6.2.2 No all-inclusive off-the-shelf solution

Participants agreed that there is no single solution that could automate patch deployments for all systems from start to finish. There are some commercially available products to manage some of the components of the patch process, but no single solution supports all software systems from all software vendors. Individual solutions may have different limitations (Hosek & Cadar, 2013). Certain vendors may provide vendor-specific solutions that work only with their software products. Microsoft’s WSUS is a good example of such a solution. The challenge for organisations that run software from multiple vendors is that it may not be feasible to run a vendors-specific solution for each vendor’s product is their environment.

6.2.3 Communication and documentation

It is noted that patch management activities are not always properly documented, and in other instances, not documented at all. Software patches can break working systems, and a lack of documentation can result in an organisation making the same mistake more than once. Liu et al. (2012) assert that some patches may introduce new vulnerabilities, however the incidents are not always recorded and linked to patch activities, making it difficult to track what was caused by patch deployments and what was not caused by patch deployments. A lack of communication and documentation could result in a scenario where the importance and risks of patch management is not well understood within the organisation. The challenge for the organisation is that patch management activities are not consistent, and with a lack of proper documentation and communication, certain patch management activities could be omitted or duplicated because no record is being kept of who is doing what.

6.2.4 Maintenance timeslots

Interviewees note that a large number of systems cannot be restarted as they need to remain available at all times, making it challenging to patch these systems. An emergency call centre systems is an example of a high availability system that could be challenging to patch. A lack of predefined maintenance slots is directly linked to the inconsistency of patch management activities. Patch management is one of various maintenance tasks, and often contend for timeslots with other maintenance tasks such as software version upgrades and antivirus definition updates. The challenge for the organisation is to balance the available timeslots between all maintenance tasks in the order of highest priority first.

6.2.5 Testing

The testing of patch deployments is noted as a challenge, largely due to available resources. Not doing enough testing could result in unexpected behaviour after patches have been deployed, while doing too much testing could introduce a delay and the vulnerabilities may be exploited before the patch is deployed. The test environment is also a challenge, because it is often not representative of the production environment, resulting in the test results being of little value. Testing extensively is time-consuming and holds the risk that a vulnerability could be exploited before the patch is deployed, while accelerating the testing holds the risk that a patch may not be tested properly and could cause problems once it is deployed. The challenge for the organisation is to ensure that the risk of testing versus the risk of not testing is understood, and also to ensure that the test environment is aligned with the production environment.

6.2.6 Legacy systems

Legacy systems pose a challenge in single and multi-vendor environments as the majority of software vendors do not provide patches or support for legacy systems. Microsoft’s Windows XP is an example of this challenge, as the support was ended in 2014, yet several individuals and organisations are still using the operating system. In the case of the organisation, certain web-based systems were initially designed to be compatible with Internet Explorer 7 (IE7) only. Windows 7 and later versions of Microsoft operating systems do not support IE7. The challenge for the organisation is to remain current with supported versions of software, while ensuring that their legacy systems (both proprietary and non-proprietary) are compatible with the newer versions of operating systems.

6.2.7 The people aspect

It is evident that not all patch management challenges are technology related. Some challenges are people problems. More specifically, it is how certain individuals view patch management.

The main reason for this type of challenge is often related to the mind-set of individuals, and preconceived conceptions such as “if it’s not broken, let’s not fix it”, which is often based on negative past experiences with patch management. The challenge for the organisation is to get individuals committed that may have had negative experiences in the past with regard to patch management.

6.3 Answering the research questions and research sub-questions

6.3.1 Research sub-questions

This section addresses the research sub-questions explicitly. The abbreviation RSQ stands for research sub-question and is used to indicate the specific research sub-question that is being addressed. For the convenience of the reader, the applicable research sub-question is presented in italics above the discussion.

6.3.1.1 Patch management challenges (RSQ 1.1)

What challenges do organisations face with regard to patch management?

The concept of patch management is simple, yet many organisations often have difficulty maintaining an active and efficient patch management strategy. Part of the reason organisations fail with patch management is because there are various challenges they are presented with, and these challenges are often not considered when patch management policies are developed. As an example, if an organisation relies on a vendor for patches, the patch management policy needs to consider the release frequency and method of the vendor as well as the time required for testing, among others. The challenges include resource constraints, testing and communication, and legacy systems. See section 6.2 for a more in-depth discussion on the patch management challenges.

6.3.1.2 Patch management challenges created in a multi-vendor software environment (RSQ 1.2)

What challenges are created by a multi-vendor environment?

In the case where software from multiple vendors is used, it becomes more challenging than in the case of a single vendor environment to manage software patches. One of the reasons for this is because there is no single comprehensive source of patch information. An organisation may have to subscribe to various newsletters to receive information regarding released patches. Although some websites specialise in getting most patch-related information in one place, it is unlikely that one website would be able to maintain this database for all software vendors in existence.

In multi-vendor environments the public disclosure of vulnerabilities by any party has a direct effect on the vulnerability of an organisation. After vulnerabilities are disclosed, attacks typically first increase before they decrease. Furthermore, the way a vulnerability is disclosed can have an effect on whether vulnerabilities are exploited or not. The reality is that patches are released by different vendors at different times using different methods. It becomes more challenging as the number of applications from different vendors increase within an organisation. The key challenge in a multi-vendor environment is that the organisation is forced to align with multiple vendors in terms of their release schedule and the format of their patch releases.

6.3.1.3 Controls to assist with patch management (RSQ 1.3)

What controls can organisations implement in order to deploy the relevant patches?

It is preferred for all stakeholders in the patch management process to be involved from the start, and the development of the patch management to be a collective effort. An all-encompassing approach could result in less resistance at a later stage where operational teams will be required to implement the technical aspects of the patch management process. Vendor-specific solutions are often not feasible due to the large number of multi-vendor applications in a given environment, and alternative approaches should be investigated.

The patch management policy should define communication and documentation requirements, as well as maintenance windows for patch deployments which are likely to result in patch management activities becoming more consistent and better documented. Periodic risk assessments should be completed and the risk of extensively testing patches versus the risk of not extensively testing patches should be understood. Effort should be taken to align the test environment with the production environment to improve the reliability of test results. The organisation should maintain current software inventory information, as this will highlight legacy applications that may require attention. Once a legacy system has been identified, the organisation could then potentially build a plan to address the legacy system. The people aspect should be kept in mind when developing the patch management problem because individuals may have a skewed perception of patch management as a result of bad past experience, and be reluctant to fully commit to patch management activities.

6.3.1.4 Patch management and prioritising (RSQ 2.1)

How do organisations prioritise patches for deployment?

Vendors often provide a prioritising guideline, but this is not always realistic for medium to large organisations. IT managers should assign priory to IT projects according to importance and in relation to available resources and infrastructure. The risk appetite of an organisation plays a crucial role in how patch management is approached and prioritised. A bank might take a more stringent approach compared to a retail company. The risk appetite will influence other factors in the patch process. The frequency of patch deployments as well as the timing and testing of patches are often in direct conflict with each other.

A patch management policy was noted as a critical tool that could aid with guidance for prioritising patch deployments within an organisation. A software baseline is used to help organisations conduct a risk analysis and provide an indication of the overall software level and landscape of the environment, while the risk analysis could assist with prioritising software patches. The information security policy in combination with the patch management policy should include criteria and guidelines for the organisation to assist with prioritising software patch deployments.

6.3.1.5 Identifying relevant software patches (RSQ 2.2)

How do organisations identify patches best suited for their security needs?

The source of patch information was noted as a concern, as software vendors currently release patches and patch-related information on their websites. Some third party services such as US Cert and CVE collate patch information into a single source. The challenge is that none of the sources cater for all software platforms, as organisations often have to subscribe to various mailing lists, websites, newsletters and news feeds for the latest patches and related information. Software applications themselves can also alert users by means of a pop-up that new updates are available; however, this is often disabled in corporate environments. Software patches from untrusted sources could contain malware or cause undesired system behaviour. For this reason it is advisable for an organisation to identify trusted sources of patch information.

An integrated patch management policy could allow organisations to distinguish between vulnerabilities that affect them versus vulnerabilities that do not affect them. The policy would also indicate what sources are considered trusted and which are not. It does not necessarily have to be a list of source names, but could be a guideline to help with the selection of a source, for example, a questionnaire to determine if a source is valid or not.

6.3.1.6 The role players in the patch management process (RSQ 2.3)

Who are the key role players in managing the patch deployment process?

Roles and responsibilities are not clearly defined within the organisation, resulting in all teams not being completely aware of what tasks they need to complete. Management is responsible to ensure that adequate resources are available to comply with the patch management policy. Management is also responsible for how timeslots are divided between various maintenance activities, and is ultimately responsible for compliance of their environments in terms of overall endpoint protection.

The patch coordinator is a representative from each environment within the organisation and is responsible for identifying all newly released patches that are applicable to the organisation. The patch coordinator must ensure patches are adequately tested before it is deployed, and also that all pre-requisites have been identified.

Software deployment administrators are responsible for administering the software deployment mechanism that deploys actual patches to the endpoints. In some instances the patch coordinator may assume this role as well, while in other cases it may be a different team. SCCM specialists generally assume this role.

Technical testers are representatives from across the IT division who are responsible for the support of certain IT systems, and are therefore able to validate that the patch has no negative impact on their specific systems. Business testers are business users that test the functionality of the applications after a patch has been deployed. This is to validate that the released patch has no negative impact on a specific system from a usability and functionality point of view.

The information security team performs an oversight, consultancy and guidance function and is accountable for the implementation and effectiveness of the patch management process. In addition to the patch coordinator, the information security team is also responsible for monitoring external sources of information around emergency patches released by vendors and/or trusted security research organisations.

6.3.2 Research questions

This section addresses the research questions explicitly. The abbreviation RQ stands for Research Question and is used to indicate the specific research question being addressed. For the convenience of the reader, the research question is presented in italics above the discussion.

6.3.2.1 The complexity with patch management (RQ1)

What causes the management of software patches to be complex?

Patch management adds a layer of complexity to an environment. It has on numerous occasions in the past caused problems with working systems. It has also on many other occasions prevented software exploitations. There are risks associated to actively applying patches, and there are risks associated to not actively applying patches. The challenge is to find the optimal balance. This section summarises the complexity of the patch management process within the company.

Patch management adds complexity to an environment in that it is something that both relies on and affects other teams. The implementation of patch management activities is done by technical staff in the operational teams, and the application of the actual software patches could affect other systems. For this reason consideration must be given to available resources and systems that could potentially be impacted.

Testing is a critical function in the patch management process. It is documented that patches have negatively affected working systems in the past. This could be due to a bug in the patch itself, the deployment mechanism, or missing prerequisites in the environment. It is however possible for a software patch to pass the testing phase and still cause a problem in the production environment. This may occur when the test environment does not reflect the production environment. Too much testing may increase the vulnerability exposure window, and too little testing could result in the patch not being properly tested. With the aforementioned in mind, testing adds a form of complexity because it is done in conjunction with available technical resources, while considering that that patches could be released at any given time.

The people aspect introduces a different set of challenges and complexities. People often may have views on patch management that may affect their behaviour. These views are formed from past experiences. A classic example is when some of the interviewees had first-hand experiences in the past of a patch breaking a working system. This experience changed their perception of software patches and a patch is now viewed as a potential threat to a working system. Another challenge is that the job role where people find themselves in may force them to act in a certain way toward patches. An information security professional and an auditor might be of the opinion that the latest patch must always be deployed as soon as possible, while the operational team may not deem patches as important, as their biggest priority is keeping the user-base operational, with or without the latest patches installed.

Users are often not familiar with the frameworks and policies that exist, and this could be as a result of a gap in the knowledge transfer process with new employees joining the organisation. This was evident when certain interviewees said there were no polices in place, yet there were. Legacy applications present a form of complexity in that these systems are often outside support agreements from software vendors, meaning no patches or support is available for these systems. Organisations may run legacy systems for a variety of reasons, including that it could be business critical, and budgetary constraints may prevent the immediate replacement of these systems. The result is that these systems may remain in production environments for prolonged time periods. Legacy systems are often vulnerable and affect the level of data security.

Documentation introduces complexity in the patch process because it is not maintained or kept in a central location. It may exist, but people requiring the documentation might not be aware of it or know where to find it.

Off-the-shelf patch management software may be able to address some of the complexities highlighted, but could also introduce other complexities. A system that seeks to automate patch deployments from end to end would still need to factor in the testing. This will require customisation of a product to be specific to the organisation in question. These tools often cost a significant amount of money, and none of the solutions support all the software applications from all the vendors on the market.

The available infrastructure within the company may have an impact on the patch management strategy of the organisation. Infrastructural limitation may add complexities as field staff could be required to update over 3G connections, incurring costly data charges. Available maintenance time slots have to be equally divided between all maintenance tasks, and this will add complexity if maintenance tasks are not properly prioritised. Other maintenance tasks may include the installation of new software, software upgrades and antivirus definition updates, and should be considered when maintenance timeslots are allocated.

6.3.2.2 Current patch management implementation within the company (RQ2)

How do organisations implement patch management in order to enhance enterprise data security?

The company adopted a patch strategy to only deploy critical and security patches. At first glance this may appear to be a safe approach. The situation could arise where a security or critical patch requires a non-security or non-critical patch as a prerequisite. This will result in a situation where the prerequisite must first be sourced, tested and deployed before the critical or security patch can be deployed. It will complicate the process, introduce a delay, and increase the vulnerability exposure time in the case when an emergency patch may need to be deployed.

The current patch management process is depicted in figure 6.1. Multiple software vendors release patches, either on a defined schedule or on an ad-hoc basis. These patches are then available to all customers with a license to use their software. In the company, the IT landscape is split into three domains: Head Office, Stores and Servers. Each of these departments would then assume the responsibility to obtain the latest patches, test them and deploy them. This process is performed at the discretion of the various department managers.

There is currently no process that defines who receives and tests these patches. Patch management processes are isolated on a company level, and may also be further isolated on a departmental level. Each of these environments has multiple decision-makers and each decision-maker can choose to deploy a software patch or not, or remain undecided. The current process does not have a mechanism in place to close the loop, and the result is an inconsistent and vulnerable environment.

The problem with the current approach is that the more decision-makers there are the more variables could exist in the process. These variables could lead to un-patched systems that could potentially be exploited, and compromise the data security of the organisation.

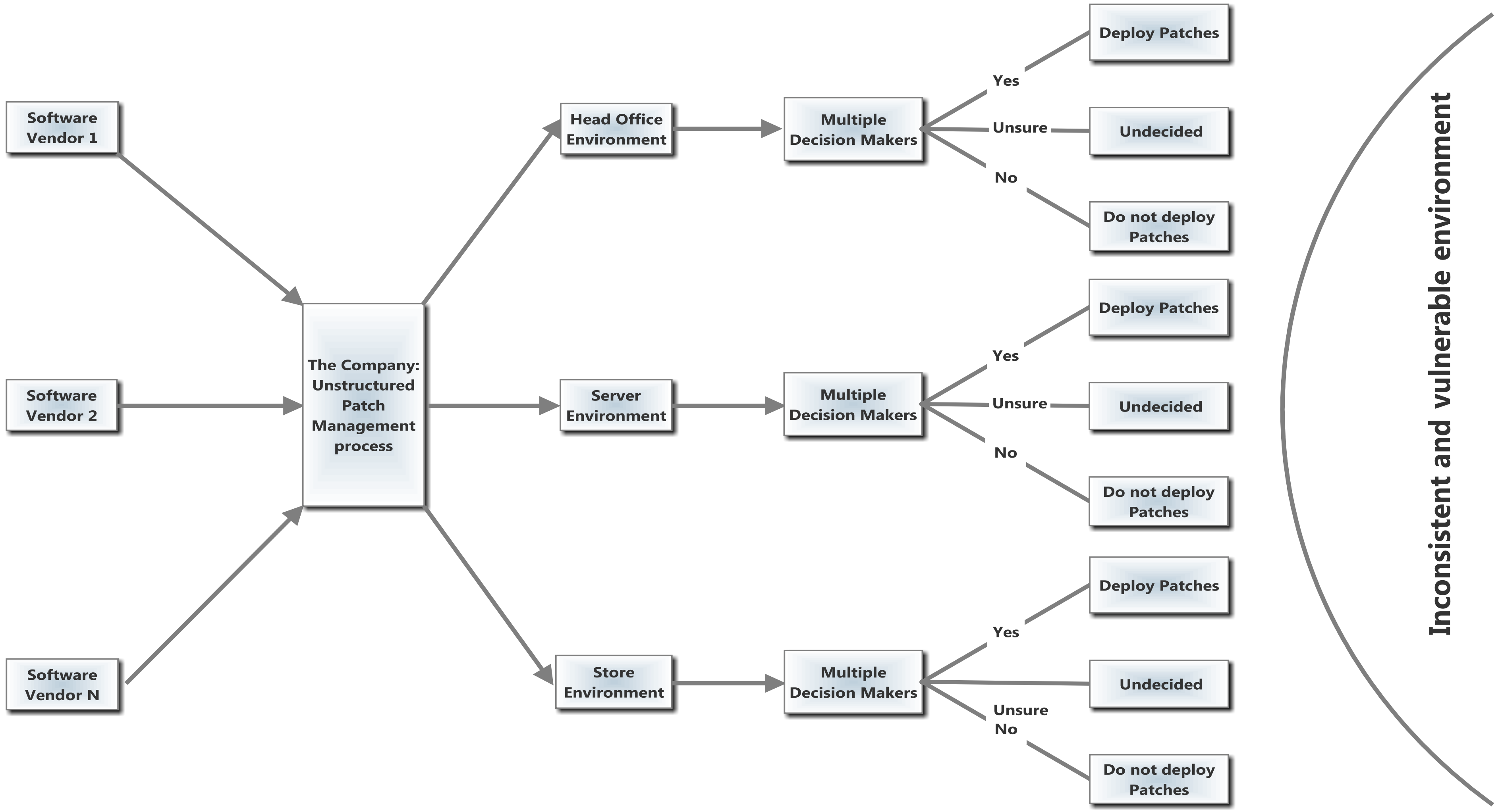


Figure 6.1: The current patch management model in the company

6.4 Proposed solution/model to the research problem

The two key problems with the current process depicted in figure 6.1 are that there are several decision-makers, and no feedback mechanism. The multiple decision-makers could introduce inconsistency, as some may decide to actively patch while others may opt not to actively patch. If there is no oversight, it would not be possible to identify outdated systems within the environment. The lack of a feedback mechanism could directly lead to an unsecure environment that is vulnerable to exploitation.

Figure 6.2 represents an improved patch management process flow, in comparison to the process depicted in figure 6.1. Figure 6.2 aims to address the two main problems highlighted in figure 6.1. Multiple vendors would still release patches as per normal, but in this proposed model, a single decision-making unit within an organisation would assume the responsibility of this function. This unit would be responsible to decide what patches should be deployed, based on the patch management policy. How these patches are deployed could remain at the discretion of the operational teams, as they would know their environment and tools best.

Once the decision-making unit has completed their function, instructions would be sent to the various implementers who would be responsible for the testing and deployment of software patches within their respective environments. Once the operations teams have completed their instructions, feedback should be sent to the information security department in the form a compliance report on a defined schedule. This will ensure that the loop is closed and that oversight is maintained on a higher level. Centralising the decision-making function could result in a more stable and secure environment that will directly increase the data security level of the organisation.

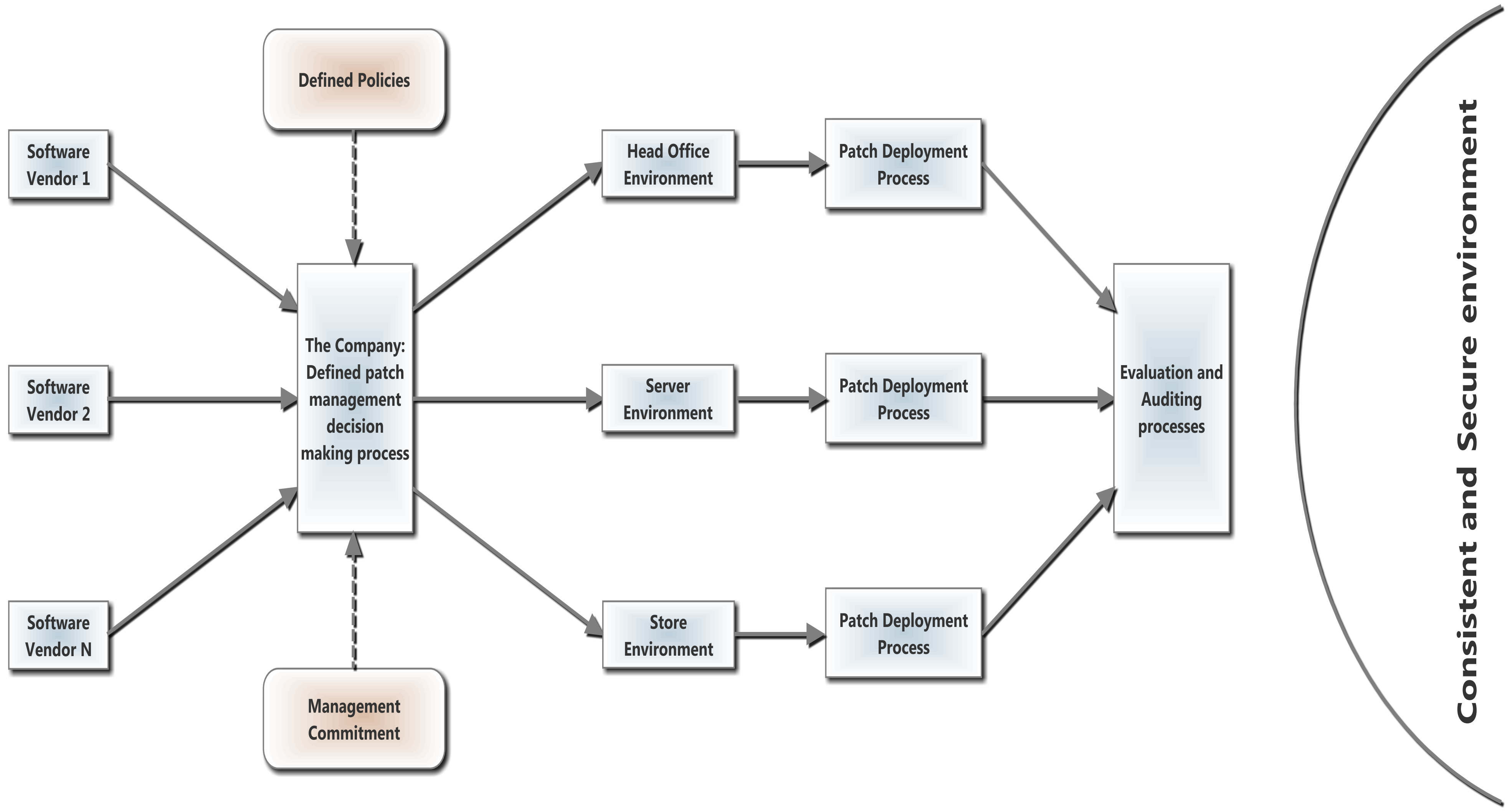


Figure 6.2: The proposed model (The Single Decision Making Model)

6.5 Recommendations

After analysing the data and considering the various challenges, as well as answering the research questions, the following recommendations are made by the author:

i. The organisation should:
   - Maintain current software inventory information of their environment
   - Revise their current approach of only deploying critical and security patches
   - Identify a tool for the distribution of software patches
   - Maintain an Exceptions Register for systems that cannot be patched due to whatever reason
   - Educate role players about the importance of patch management, and the consequence of failing to properly manage patches

ii. The patch management policy should:
   - Define the roles and responsibilities in a clear and concise manner
   - Be aligned to the available resources and infrastructure of the organisation
   - Provide guidance in terms of trusted sources for patches and related information
   - Be developed in a combined effort and all stakeholders being consulted
   - Define the overall test approach within the organisation and identify the technical tester and business testers
   - Define the criteria and requirements for compensating controls in cases where regular patching is not possible
   - Be endorsed by senior management

iii. Test environments should be representative of the production environment and all patch testing should be signed off by the appropriate stakeholders before patches are deployed.

iv. Patch management activities should be consistent and include complete and relevant documentation.

v. Suitable maintenance windows should be identified and divided equally across all maintenance tasks.

vi. Prerequisites and tasks should be identified and completed upfront in order to ensure that there are no unforeseen events during the patch deployments.

vii. The number of decision-makers in the patch management process should be reduced and tasked to management.

viii. Regular vulnerability scans should be conducted to identify vulnerable systems.

ix. The communication process should be consistent, well-defined and bi-directional between organisations and software vendors.

x. All patch deployment plans should include roll-back procedures in order to revert back to the previous configuration if any problems arise during and after the deployment of a software patch.

xi. The base template of the operating system should be updated with the latest patches to minimise the future work effort. This will ensure that any deployed system or software contains the required patches at the time of deployment, and reduce the windows of exposure.

xii. The deployment success rate should be reviewed to ensure that the deployment has reached all intended target devices.

6.6 Conclusion

The research presents a case study on a Cape Town-based retailer that is representative of the research problem, and proposes a model to help organisations implement a more efficient patch management process and simultaneously improve data security. The objectives of the study are to identify some of the challenges organisations face when implementing patch management, determine the challenges faced by organisations with multi-vendor systems, and determine how patch management affects enterprise data security. The objectives of the study are adequately addressed and a brief synopsis of the findings and importance thereof is presented below.

Legacy systems often fall outside support agreements with vendors and therefore do not receive software patches or support from vendors. The result is that legacy systems in use become vulnerable and could be exploited. The level of data security is affected as vulnerable systems could result in a data breach and have serious consequences for organisations.

Limited resources restrict the organisation’s ability to maintain an effective patch management strategy. The limitation of resources could hinder the organisation’s ability to implement an efficient patch management policy, and therefore result in the organisation being vulnerable to exploitation and data breaches. The organisation stores customer records and personally identifiable information that could be compromised if systems are compromised.

No single documented source of patch information could be found that provides information and software patches for all software applications from all software vendors. For this reason, organisations are often forced to subscribe to various websites and newsletters. An organisation could potentially miss a critical or security update from a vendor and remain vulnerable even after a vulnerability has been disclosed and the patch has been released.

People challenges relate to the mind-set of role players and are often formed by past experiences. If a role-player had a negative experience in the past in terms of patch management, they could develop a negative attitude toward patch management and reason that ‘if it’s not broken, don’t fix it’. If all role players are not aligned regarding the importance of patch management, it could hinder the patch management process as role players would not actively be implementing patch management.

There are multiple decision-makers in the current patch management process. This could result in inconsistencies within the environment, as certain individuals may decide one course of action, while other individuals may decide on a different course of action. The ultimate result of the current model is that certain systems may remain unpatched for long periods of time, or indefinitely, leaving the organisation at risk of exploitation.

There is a lack of a proper communication mechanism. Certain role players may or may not complete tasks, but do not communicate this with other role players. These tasks could then either be omitted or duplicated, as decisions are based on assumptions. This has a direct effect on the resources of the organisation.

Organisations are often forced to align with software vendors. Organisations can only test and deploy patches once the vendor has released it. If vulnerabilities are disclosed and software vendors do not have a patch available at the time, it leaves the organisation vulnerable until the vendor provides a patch.

The organisation's current approach is to only deploy critical and security patches. A critical or security patch may have a non-critical or non-security patch as a prerequisite. This could introduce undesired delays when a critical or security patch needs to be deployed as the prerequisite would first have to be sourced, tested and deployed before the security or critical patch can be deployed.

The test environment often does not represent the production environment. The test results will not be of value, as patches may have a different effect in the production environment. Resources will be wasted on testing if the test environment is not representative of the production environment, as the test results will be of little value.

The testing of software patches is time-consuming and resource-intensive when it is done extensively. Organisations may opt not to test extensively in the interest of saving costs. If patches are not tested extensively, it could result in deployment and/or functionality problems once it has been deployed, and negatively affect the perception around patch management.

The aim of the study is to explore the complexities of software patch management in order to enhance enterprise data security within organisations. The study concludes that organisations are presented with various challenges (section 6.2) that make it difficult for organisations to manage an active and effective patch management strategy. As a result, systems are often not patched in a timely manner, or at all, leaving them vulnerable to exploitation and therefore compromise the level of data security within an enterprise. For this reason it is crucial for organisations to be aware of these challenges when planning a patch management strategy.

6.7 Future research

Future research opportunities could do the exact same study at different organisations and compare the results. If the results are similar, it would be a step towards proving generalisability for the study. If the results differ greatly, it would be interesting to understand why.

Being able to quantify the return on investment in terms of the monetary value that patch management can add, will go a long way towards obtaining buy-in from management. This was identified as one of the challenges.

Testing is one of the single biggest challenges that emerged and special focus on patch testing could add value. A study could consider the importance of aligning a test environment with the production environment, and in so doing could ensure the delivery of more useful and accurate test results.

Another challenge was the people aspect. Perhaps a psychological approach, rather than a technological approach, could address the human aspect of the problem. People's past experiences in terms of patch management create preconceived notions, and this affects their decision-making in present situations. An investigation is needed on how to promote objectivity in terms of patch management.

REFERENCES

Ackling, T., Alexander, B. & Grunert, I. 2011. Evolving patches for software repair. *Proceedings.* The 13th Annual Conference on Genetic and Evolutionary Computation, Dublin, 12-16 July.

Ahmad, A., Maynard, S.B. & Park, S. 2014. Information security strategies: towards an organisational multi-strategy perspective. *Journal of Intelligent Manufacturing,* 25(2):357-70.

Albanese, M., Jajodia, S., Singhal, A. & Wang, L. 2013. An efficient approach to assessing the risk of zero-day vulnerabilities. *Proceedings.* The 10th International Conference on Security and Cryptography, Reykjavík, 29-31 July.

AlEroud, A. & Karabatis, G. 2012. A contextual anomaly detection approach to discover zero-day attacks. Paper presented at the 2012 International Conference on Cyber Security (CyberSecurity), Alexandria, Virginia, 14-16 December.

Arena, M., Arnaboldi, M. & Azzone, G. 2011. Is enterprise risk management real? *Journal of Risk Research,* 14(7):779-97.

Arora, A., Krishnan, R., Telang, R. & Yang, Y. 2010. An empirical analysis of software vendors' patch release behavior: impact of vulnerability disclosure. *Information Systems Research,* 21(1):115-32.

Arora, A., Telang, R. & Xu, H. 2008. Optimal policy for software vulnerability disclosure. *Management Science,* 54(4):642-56.

August, T. & Tunca, T.I. 2011. Who should be responsible for software security? A comparative analysis of liability policies in network environments. *Management Science,* 57(5):934-59.

BBC News. 2015. *Android bug: MMS threat affects 'one billion' phones*. Available: http://www.bbc.com/news/technology-33689399. [Accessed: 9 August 2015].

Beasley, M.S., Clune, R. & Hermanson, D.R. 2005. Enterprise risk management: an empirical analysis of factors associated with the extent of implementation. *Journal of Accounting and Public Policy,* 24(6):521-531.

Begel, A., Khoo, Y.P. & Zimmermann, T. 2010. Codebook: discovering and exploiting relationships in software repositories. Paper presented at the Software Engineering, 2010 ACM/IEEE 32nd International Conference, Cape Town, 1-8 May.

Benbasat, I., Goldstein, D.K. & Mead, M. 1987. The case research strategy in studies of information systems. *MIS Quarterly,* 11(3):369-86.

Beres, Y. & Griffin, J. 2012. Optimizing network patching policy decisions. *In Information Security and Privacy Research*. Berlin, Heidelberg: Springer: 424-442.

Berg, B.L. & Lune, H. 2004. *Qualitative research methods for the social sciences.* Boston, MA: Pearson.

Bilge, L. & Dumitras, T. 2012. Before we knew it: an empirical study of zero-day attacks in the real world. *Proceedings*. The 2012 ACM conference on Computer and communications security, Raleigh, NC, 16-18 October.

Blumberg, B., Cooper, D.R. & Schindler P.S. 2005. *Business research methods.* 2nd ed. Berkshire: McGraw-Hill.

Bock, T. & Sergeant, J. 2002. Small sample market research. *International Journal of Market Research*, 44(2):235.

Bryman, A. 2006. Integrating quantitative and qualitative research: how is it done? *Qualitative Research,* 6(1):97-113.

Burrell, G. & Morgan, G. 1979. *Sociological paradigms and organisational analysis.* London: Heinemann.

Capriz, M. 2011. *Ben Franklin and the internet snoops*. Available: http://www.academia.edu/6712729/Internet_Snoops_2. [Accessed: 14 July 2016].

Cárdenas, A.A., Amin, S., Lin, Z.S., Huang, Y.L., Huang, C.Y. & Sastry, S. 2011. Attacks against process control systems: risk assessment, detection, and response. *Proceedings*. The 6th ACM symposium on Information, Computer and Communications Security, Hong Kong, 22-24 March.

Cavusoglu, H., Cavusoglu, H. & Zhang, J. 2008. Security patch management: share the burden or share the damage? *Management Science,* 54(4):657-70.

Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K. et al. 2011. Comprehensive experimental analyses of automotive attack surfaces. Paper presented at the 20th USENIX Security Symposium, San Francisco, CA, 8-12 August.

Chen, X., Mao, Y., Mao, Z.M. & Van der Merwe, J. 2010. Declarative configuration management for complex and dynamic networks. Paper presented at the 6th International Conference on emerging Networking EXperiments and Technologies (CoNEXT), Philadelphia, 30 November-3 December.

Choi, J.P., Fershtman, C. & Gandal, N. 2010. Network security: vulnerabilities and disclosure policy. *The Journal of Industrial Economics,* 58(4):868-94.

Chow, W.S. & On Ha, W. 2009. Determinants of the critical success factor of disaster recovery planning for Information Systems. *Information Management and Computer Security* 17(3):248-75.

Cohen, D. & Crabtree, B. 2006. *Qualitative research guidelines project.* Princeton: Robert Wood Johnson Foundation.

Cronjé, J.C. 2012. The abc (aim, belief, concern) instant research question generator. Unpublished Manuscript.

Davidson, S. 2013. Privacy through obscurity. *IEEE Design and Test,* 30(5):96.

Denzin, N. & Lincoln, Y. (eds). 2000. *Handbook of qualitative research*. 2nd ed. Thousand Oaks: Sage: 413-427.

Dubois, A. & Gadde, L.E. 2002. Systematic combining: an abductive approach to case research. *Journal of Business Research,* 55(7):553-60.

Ehlert, S., Geneiatakis, D. & Magedanz, T. 2010. Survey of network security systems to counter sip-based denial-of-service attacks. *Computers and Security,* 29(2):225-43.

Eisenhardt, K.M. 1989. Building theories from case study research. *Academy of Management Review,* 14(3):532-50.

Elky, S. 2006. An introduction to information system risk management. Available: https://www.sans.org/reading-room/whitepapers/auditing/introduction-information-system-risk-management-1204. [Accessed: 24 July 2016].

Farahmand, F., Navathe, S.B., Enslow, P.H. & Sharp, G.P. 2003. Managing vulnerabilities of information systems to security incidents. Paper presented at the 5th International Conference on Electronic Commerce, Pittsburgh, Pennsylvania*,* 30 September-3 October.

Farwick, M., Agreiter, B., Breu, R., Ryll, S., Voges, K. & Hanschke, I. 2011. Automation processes for enterprise architecture management. Paper presented at the 2011 15th IEEE International Enterprise Distributed Object Computing Conference Workshops, Helsinki, 29 August-2 September.

Felmetsger, V., Cavedon, L., Kruegel, C. & Vigna, G. 2010. Toward automated detection of logic vulnerabilities in web applications. Paper presented at the 19th USENIX Security Symposium, Washington, DC, 11-13 August.

Flyvbjerg, B. 2006. Five misunderstandings about case-study research. *Qualitative Inquiry,* 12(2):219-45.

Foley, S.N. 2009. Security risk management using internal controls. *Proceedings*. The *first ACM Workshop on Information Security Governance, New York*, 9-13 November.

Fossey, E., Harvey, C., McDermott, F. & Davidson, L. 2002. Understanding and evaluating qualitative research. *Australian and New Zealand Journal of Psychiatry,* 36(6):717-32.

Fossi, M., Egan, G., Haley, K., Johnson, E., Mack, T., Adams, T., Blackbird, J. et al. 2011. *Symantec Global Internet Security Threat Report: Trends for 2010.* Vol XV, Symantec: 1-20.

French, A.M. 2012. A case study on e-banking security—when security becomes too sophisticated for the user to access their information. *Journal of Internet Banking and Commerce,* 17(2):1-14.

Gerace, T. & Cavusoglu, H. 2009. The critical elements of the patch management process. *Communications of the ACM,* 52(8):117-21.

Gerring, J. 2004. What is a case study and what is it good for? *American Political Science Review,* 98(2):341-54.

Gibson, D. 2014. *Comptia Security+: get certified get ahead*. United States: Ycda LLC: 2-165.

Giuffrida, C., Kuijsten, A. & Tanenbaum, A.S. 2013. Safe and automatic live update for operating systems*. ACM SIGPLAN Notices*, 48(4):279-292

Gogoi, P., Bhattacharyya, D.K., Borah, B. & Kalita, J.K. 2011. A survey of outlier detection methods in network anomaly identification. *The Computer Journal,* 54(4):570-88.

Gollman, D. 2011. *Computer Security*. 3rd ed. United Kingdom: Wiley: 178.

Gonzalez, R. & Locasto, M. 2013. *Classifying the data semantics of patches*. Technical Report 2013-1047-14, University of Calgary.

Gummesson, E. 2007. Case study research and network theory: birds of a feather. *Qualitative Research in Organisations and Management: An International Journal,* 2(3):226-48.

Highsmith, J.A. 2013. *Adaptive software development: a collaborative approach to managing complex systems.* New York: Dorset House.

Hirschheim, R. & Klein, H.K. 1989. Four paradigms of information systems development. *Communications of the ACM*, *32*(10):1199-1216.

Holloway, I. & Wheeler, S. 2013. *Qualitative research in nursing and healthcare.* United Kingdom: John Wiley & Sons.

Höne, K. & Eloff, J.H.P. 2002. Information security policy—what do international information security standards say? *Computers & Security*, 21(5):402-409.

Hosek, P. & Cadar, C. 2013. Safe software updates via multi-version execution. *Proceedings.* The 2013 International Conference on Software Engineering, May. IEEE Press: 612-621.

Huang, H., Baset, S., Tang, C., Gupta, A., Sudhan, K.N.M., Feroze, F., Garg, R. et al. 2012. Patch management automation for enterprise cloud. Paper presented at the 2012 IEEE Network Operations and Management Symposium [front matter], Maui, HI, 16-20 April.

Ioannidis, C., Pym, D. & Williams, J. 2012. Information security trade-offs and optimal patching policies. *European Journal of Operational Research,* 216(2):434-44.

Jansen, W.A. 2011. Cloud hooks: security and privacy issues in cloud computing. Paper presented at the 2011 44th Hawaii International Conference on System Sciences (HICSS), Kauai, HI, 4-7 January.

Jenkins, D., Arnaud, J., Thompson, S., Yau, M. & Wright, J. 2014. Version control and patch management of protection and automation systems. Paper presented at the 2014 12th International Conference on Developments in Power System Protection (DPSP), Copenhagen, Denmark, 31 March-3 April.

Johnson, R.B. & Onwuegbuzie, A.J. 2004. Mixed methods research: a research paradigm whose time has come. *Educational Researcher,* 33(7):14-26.

Joshi, A.P. 2013*. Linked data for software security concepts and vulnerability descriptions.* Maryland University*.*

Kawulich, B.B. 2005. Participant observation as a data collection method. *FQS Forum: Qualitative Sozialforschung/Forum Qualitative Social Research,* 6(2), Art. 43.

Kermani, M.M., Zhang, M., Raghunathan, A. & Jha, N.K. 2013. Emerging frontiers in embedded security. Paper presented at the 26th International Conference on VLSI Design, Pune, 5-10 January.

King, N. 1994. The qualitative research interview. *In* Cassel, C. & Symon, G. (eds). *Qualitative methods in organisational research: a practical guide.* London: Sage: 14-36.

Kitzinger, J. 1994. The methodology of focus groups: the importance of interaction between research participants. *Sociology of Health and Illness,* 16(1):103-21.

Kouns, J. & Minoli, D. 2011. *Information technology risk management in enterprise environments: a review of industry practices and a practical guide to risk management teams.* Hoboken, NJ: John Wiley & Sons.

Kurose, F. & Ross, K. 2010. *Computer Networking*. 5th ed. Boston: Pearson: 424.

Lahtela, A. & Jäntti, M. 2011. Challenges and problems in release management process: a case study. Paper presented at the 2011 IEEE 2nd International Conference on Software Engineering and Service Science (ICSESS), Beijing, 15-17 July.

Lahtela, A., Jäntti, M. & Kaukola, J. 2010. Implementing an ITIL-based IT service management measurement system. Paper presented at the Fourth International Conference on Digital Society ICDS 2010, St. Maarten, 10-16 February.

Le Goues, C., Forrest, S. & Weimer, W. 2013. Current challenges in automatic software repair. *Software Quality Journal,* 21(3):421-43.

Le Goues, C., Nguyen, T., Forrest, S. & Weimer, W. 2012. Genprog: A generic method for automatic software repair. *IEEE Transactions on Software Engineering*, *38*(1):54-72.

Leavitt, N. 2011. Internet security under attack: the undermining of digital certificates. *Computer,* 44(12):17-20.

Liu, B., Shi, L., Cai, Z. & Li, M. 2012. Software vulnerability discovery techniques: a survey. Paper presented at the 2012 4th International Conference on Multimedia Information Networking and Security (MINES), Nanjing, 2-4 November.

Long, J. 2011. *No tech hacking: a guide to social engineering, dumpster diving, and shoulder surfing.* Burlington, MA: Syngress.

Marshall, M.N. 1996. Sampling for qualitative research. *Family Practice,* 13(6):522-5.

McNaughton, B., Ray, P. & Lewis, L. 2010. Designing an evaluation framework for IT service management. *Information and Management,* 47(4):219-25.

McQueen, M., Boyer, W., McQueen, T. & McBride, S. 2009. Empirical estimates of 0day vulnerabilities in control systems. Paper presented at the SCADA Security Scientific Symposium 2010. 6-9 April.

Mell, P., Bergeron, T. & Henning, D. 2005. Creating a patch and vulnerability management program. *National Institute of Standards and Technology Special Publication,* 800:40.

Mell, P.K. & Scarfone, S. 2007. Guide to intrusion detection and prevention systems (IDPS*). NIST special publication,* 800(2007):94.

Mell, P., Scarfone, K. & Romanosky, S. 2006. Common vulnerability scoring system. *IEEE Security and Privacy,* 4(6):85-9.

Mell, P., Scarfone, K. & Romanosky, S. 2007. *A complete guide to the common vulnerability scoring system version 2.0.* FIRST-Forum of Incident Response and Security Teams.

Meyer, C.B. 2001. A case in case study methodology. *Field Methods,* 13(4):329-52.

Microsoft. 2008. *Help protect yourself from the Conficker worm*. Available: https://www.microsoft.com/en-us/safety/pc-security/conficker.aspx. [Accessed: 24 May 2016].

Mohammadi, H. 2013. A systems engineering framework for implementing a security and critical patch management process in diverse environments (academic departments' workstations). *Journal of Information Technology Management,* XXIV(4):51-61.

Mohr, S. & Rahman, S.S. 2011. IT security issues within the video game industry. *International Journal of Computer Science and Information Technology,* 3(5):1-16.

Moore, D. & Shannon, C. 2002. Code-Red: a case study on the spread and victims of an Internet worm. *Proceedings.* The 2nd ACM SIGCOMM Workshop on Internet measurement. ACM: 273-284.

Morgan, G. &. Smircich, L. 1980. The case for qualitative research. *Academy of Management Review,* 5(4):491-500.

Myers, M.D. 1997. Qualitative research in information systems. *Management Information Systems Quarterly,* 21(2):241-2.

Naik, K. & Tripathy, P. 2011. *Software testing and quality assurance: theory and practice.* Hoboken, NJ: John Wiley & Sons.

Neuman, W.L. 2011. Social research methods: quantitative and qualitative approaches. Boston*:* Pearson: 9-65.

Nicastro, F.M. 2011. *Security patch management.* CRC Press.

Noor, K.B.M. 2008. Case study: a strategic research methodology. *American Journal of Applied Sciences,* 5(11):1602-4.

Okamura, H., Tokuzane, M. & Dohi, T. 2009. Optimal security patch release timing under non-homogeneous vulnerability-discovery processes. Paper presented at the ISSRE 2009 20th International Symposium on Software Reliability Engineering, Karnatakam, 16-19 November.

Oracle. 2013. *Oracle critical patch update advisory - October 2013.* Available: http://www.oracle.com/technetwork/topics/security/cpuoct2013-1899837.html. [Accessed: 24 May 2016].

Panzica La Manna, V. 2011. Dynamic software update for component-based distributed systems. Paper presented at the 16th International Workshop on Component-Oriented Programming, Boulder*,* CO, 20-24 June.

Patnayakuni, R. & Patnayakuni, N. 2014. Information security in value chains: a governance perspective. Paper presented at the Twentieth Americas Conference on Information Systems, Savannah, 2014, Savannah, 7-9 August.

Payer, M. & Gross, T.R. 2013. Hot-patching a web server: a case study of asap code repair. Paper presented at the 2013 Eleventh Annual Conference on Privacy, Security and Trust, Catalonia*,* 10-12 July.

Qian, Z., Mao, Z.M., Rayes, A. & Jaffe, D. 2011. Designing scalable and effective decision support for mitigating attacks in large enterprise networks. In *Security and privacy in communication networks*, Heidelberg: Springer: 1-18.

Raja, U. & Tretter, M.J. 2011. Classification of software patches: a text mining approach. *Journal of Software Maintenance and Evolution: Research and Practice,* 23(2):69-87.

Ramaswamy, A., Bratus, S., Smith, S.W. & Locasto, M.E. 2010. Katana: a hot patching framework for elf executables. Paper presented at the Fifth International Conference on Availability, Reliability, and Security ARES 2010, Krakow, 15-18 February.

Ransbotham, S. & Mitra, S. 2013. *The impact of immediate disclosure on attack diffusion and volume in Economics of information security and privacy III*. New York: Springer: 1-12.

Rass, S. 2014. Complexity of network design for private communication and the p-vs-np question. *International Journal of Advanced Computer Science and Applications,* 5(2):148-57.

Rinard, M. 2011. Manipulating program functionality to eliminate security vulnerabilities. In *Moving target defense.* New York: Springer: 109-115.

Ritchie, J., Lewis, J., Nicholls, C. M. & Ormston, R. 2013. *Qualitative research practice: a guide for social science students and researchers.* London: Sage.

Ryan, G.W. & Bernard, H.R. 2003. Techniques to identify themes. *Field Methods,* 15(1):85-109.

Sabahi, F. 2011. Cloud computing security threats and responses. Paper presented at the 2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN), Xi'an, 27-29 May.

Saleem, S., Yu, Y. & Nuseibeh, B. 2012. An empirical study of security requirements in planning bug fixes for an open source software project. The Open University, United Kingdom. ISSN: 1744-1986.

Scandariato, R., Wuyts, K., & Joosen, W. 2015. A descriptive study of Microsoft's threat modeling technique. *Requirements Engineering* 20(2):163-80.

Schryen, G. 2011. Is open source security a myth? *Communications of the ACM,* 54(5):130-40.

Sethanandha, B.D. 2011. Improving open source software patch contribution process: methods and tools. Paper presented at the 33rd International Conference on Software Engineering (ICSE), Waikiki*,* 21-28 May.

Shahriar, H. & Zulkernine, M. 2012. Mitigating program security vulnerabilities: approaches and challenges. *ACM Computing Surveys (CSUR)*, 44(3):11.

Sharma, A., Kalbarczyk, Z., Barlow, J. & Iyer, R. 2011. Analysis of security data from a large computing organisation. Paper presented at the 2011 IEEE/IFIP 41st International Conference on Dependable Systems & Networks (DSN), Hong Kong, 27-30 June.

Shavelson, R.J., Phillips, D.C., Towne, L. & Feuer, M.J. 2003. On the science of education design studies. *Educational Researcher,* 32(1):25-8.

Sihvonen, H.M. & Jäntti, M. 2010. Improving release and patch management processes: an empirical case study on process challenges. Paper presented at the Fifth International Conference on Software Engineering Advances ICSEA 2010, Nice*,* 22-27 August.

Singhal, A. & Ou, X. 2011. *Security risk analysis of enterprise networks using probabilistic attack graphs.* Gaithersburg: Citeseer.

Sommerville, I. 2011. *Software Engineering*. 9th ed. Boston: Pearson: 15-580.

Sotirov, A. 2006. Hotpatching and the rise of third-party patches. Paper presented at the Hat USA 2006 Briefing and Training, Las Vegas*,* 29 July-3 August.

Souppaya, M. & Scarfone, K. 2013. *Guide to Enterprise Patch Management Technologies,* Vol. 800. Gaithersburg: National Institute of Standards and Technology.

Stake, R.E. 2013. *Multiple case study analysis.* New York, NY: Guilford Press.

Stolikj, M., Cuijpers, P.J. & Lukkien, J.J. 2013. Patching a patch-software updates using horisontal patching. *IEEE Transactions on Consumer Electronics,* 59(2):435-41.

Subashini, S. & Kavitha, V. 2011. A survey on security issues in service delivery models of cloud computing. *Journal of Network and Computer Applications,* 34(1):1-11.

Szefer, J., Keller, E., Lee, R.B. & Rexford, J. 2011. Eliminating the hypervisor attack surface for a more secure cloud. Paper presented at the 18th ACM conference on Computer and communications security, Switzerland*,* 17-21 October.

Thakare, S.V. & Gore, D.V. 2014. Comparative study of cia and revised-cia algorithm. Paper presented at the 2014 Fourth International Conference on Communication Systems and Network Technologies, Bhopal, 7-9 April.

Torres, J.M., Sarriegi, J.M., Santos, J. & Serrano, N., 2006. Managing information systems security: critical success factors and indicators to measure effectiveness. *In Information Security*. Berlin, Heidelberg: Springer: 530-545.

Tutt, A. 2014. Aftermarketfailure: Windows XP's end of support. *Michigan Law Review First Impressions,* 112:109-16.

Vaismoradi, M., Turunen, H. & Bondas, T. 2013. Content analysis and thematic analysis: implications for conducting a qualitative descriptive study. *Nursing & Health Sciences*, 15(3):398-405.

Wang, L., Jajodia, S., Singhal, A. & Noel, S. (eds). 2010. K-zero day safety: measuring the security risk of networks against unknown attacks. *In European Symposium on Research in Computer Security.* Berlin, Heidelberg: Springer: 573-587.

Wei, D., Lu, Y., Jafari, M., Skare, P.M. & Rohde, K. 2011. Protecting smart grid automation systems against cyberattacks. *IEEE Transactions on Smart Grid,* 2(4):782-95.

Welman, J.C., Kruger, F. & Mitchell, B. 2005. *Research methodology.* 3rd ed. Cape Town: Oxford University Press.

Wright, J.L. 2014. *Software vulnerabilities: lifespans, metrics, and case study.* University of Idaho, Moscow, Idaho.

Xu, J. & Croft, W.B. 1996. Query expansion using local and global document analysis. Paper presented at the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Zurich, 18-22 August.

Yang, B., Zeng, S., Ayachitula, N., & Puri, R. 2011. SLA-driven applicability analysis for patch management. Paper presented at the 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011), Dublin, 23-27 May.

Yin, R.K. 1981. The case study crisis: some answers. *Administrative Science Quarterly,* 26(1):58-65.

Yin, R.K. 2013. C*ase study research: design and methods.* Thousand Oaks: Sage.

Yu, J., Han, J., Schneider, J.G., Hine, C. & Versteeg, S. 2012. A virtual deployment testing environment for enterprise software systems. *Proceedings.* The 8th International ACM SIGSOFT Conference on Quality of Software Architectures, June. ACM: 101-110.

Zhao, D., Furnell, M. & Al-Ayed, A. 2009. The research on a patch management system for enterprise vulnerability update. Paper presented at the 2009 International Conference on Information Engineering – ICIE 2009, Taiyuan, 10-11 July.

Zhou, C.V., Leckie, C. & Karunasekera, S. 2010. A survey of coordinated attacks and collaborative intrusion detection. *Computers and Security* 29(1):124-40.

Zhu, Q., McQueen, M., Rieger, C. & Basar, T. 2011. Management of control system information security: control system patch management. *Proceedings*. The Workshop on the Foundations of Dependable and Secure Cyber-Physical Systems (FDSCPS-11), Chicago, IL*,* 11 April.

Zseby, T., King, A., Brownlee, N. & Claffy, K. C. 2013. *The day after patch Tuesday: effects observable in IP darkspace traffic in passive and active measurement.* Heidelberg: Springer: 273-275.

APPENDIX A: THREE INTERVIEW TRANSCRIPTS

(All transcripts and recordings available on request)

TRANSCRIPT NO 1 – SECURITY SPECIALIST

1. What is your current job title and what previous job roles did you perform before you started your current job? (Previous companies and job roles)

Interviewee: Okay, my current job title is the IT Governance and Risk Specialist…

Interviewer: Okay

Interviewee: …and I won't give you my complete history but I covered two previous roles; before this role I was the Information Security Specialist at Metropolitan life and before that I was IT systems administrator at Ackermans.

Interviewer: Okay

Interviewee: Ya and that, as you can see it was more general earlier back and then as years went on it became more specialised and within the context of Patch Management I dealt with those things prior to this current role. My current role is more documentation of the whole process and how it should work [unclear]

Interviewer: Okay

2. How familiar are you with the generic issues in Patch Management? (In general, not company specific)

Interviewee: Quite familiar, luckily with LinkedIn and all these internet based web services one gets a good idea of how the issues across the world actually and actually last week I was participating in a question that someone posted of Patch management issues within small to medium businesses and, ja, so I'm…

Interviewer: You're fairly up to date.

3. To what extent is patch management part of your job (What percentage of your time is spent on patch management functions)?

Interviewee: Less than five percent.

Interviewer: Less than five percent realistically speaking, okay.

4. How familiar are you with your company's patch management issues? (What are some of the current issues?)

Interviewee: Quite familiar actually because I've actually drafted the Patch Management process document so I had a few sessions with different stakeholders so I got some insight as to who does what and what's currently happening.

Interviewer: Okay that's actually my next question

5. Are you aware who the role players are in the patch management process? (Who is currently doing what?)

Interviewee: Indeed

6. Who do you think should be added or removed from the patch management process?

Interviewee: Well like I said, It's a different situation in each company but if I, ja, I would say that there should be people removed from the process, currently, uhm, specifically in the store area because I feel that there isn't a the expertise or they don't have access to the resources that is currently being used within the other HO domain so let me put it plainly: the people that's doing the store patching, that's not the same people that's doing the rest of the unit…

Interviewer: Okay

Interviewee: That's how I currently know the situation.

Interviewer: So there's no central point where it feeds down from?

Interviewee: Correct, that is something I would like to change.

7. What are the problems associated with managing patches from any external vendors? (What are the challenges when acquiring Microsoft or Adobe patches?)

Interviewee: Generally when acquiring patches, there is not much challenges. For the most part it is seamless.

8. Are you aware of any test procedure for patches before they are deployed? (What is the current procedure? What do you think should the procedure be?)

Interviewee: Yes I'm aware of it, but in the same breath I've also established those testing procedures within the document that I set…

Interviewer: So it is established?

Interviewee: It's not established it's, err, it's drafted.

Interviewer: Okay

Interviewee: So it's not yet been followed.

9. How are patches prioritised within the organisation? (Do you have any prioritisation policy in place or do you rely on that provided by software vendors?) (Interviewer made an e.g. of Microsoft)

Interviewee: We have our own; it's documented in our draft Patch Management Policy which has also been incorporated into a process document that I'm busy working on.

Interviewer: Okay

10. **How are patches acquired from different vendors? (How do you know what is available? What mechanism do you use to acquire these patches? How do you think this process can be improved?)**

Interviewee: Manually as well as automated, in some instances it's being automatically imported into WSUS in other ways or like in the other domains server domain the guys are manually downloading their Patches to be deployed.

Interviewer: And do we know who's downloading what?

Interviewee: Like I said, uhm, it's both manual and automated up until the process gets applied.

Interviewer: Okay

Interviewee: The document I was talking about, so once that has been applied then we will know exactly who's doing what, but currently I have a rough idea of who's getting it automatically and who's manually, a rough idea.

Interviewer: Okay and tell me the frequency of applying these patches.

Interviewee: Each system's owner is currently doing it the way they perceive necessary up until the policy and the document is being signed off.

Interviewer: Okay

11. **Based on past experience, what incidents occurred as a direct result of failing to properly manage patches? (Malware infections? Security breaches? Integrity compromised? Confidentiality compromised?)**

Interviewee: The most frequent and visible sign is a worm-breakout when, a, when you have a worm breakout on the network you can easily see which systems or work stations or which servers is the most affected and those are the ones typically that's the worst patched.

Interviewer: Okay

Interviewee: As well as system crashes like instability within systems or they can't integrate with other newer systems because they haven't been updated, so those are the two frequent sources or causes of systems not being patched.

Interviewer: Okay

12. What components would you like to see go into a patch management policy? (What should be defined in the policy?)

Interviewee: Uhm…you said who should develop the policy?

Interviewer: Uhm, number 12.

Interviewee: Oh sorry! Oh ya, uhm, the scope, what should be patched anything ranging from business applications to the third party applications, expectations of patch management, how often and you know all the expectations that goes around patch management with as well as the schedule and then the roles and responsibilities - who should be patching and who should know about the patching.

Interviewer: Okay

13. Who should develop the policy, implement the policy, and police the policy?

Interviewee: Information security

Interviewer: Information security development and then obviously the implementation of that policy should be delegated.

Interviewee: Yes, the implementation thereof should be delegated to the relevant people that's responsible for the system, the system owners basically, the ones responsible for managing the tool that deploys the patches.

Interviewer: And then who should police this policy? You develop it, you delegate it, who should make sure it's actually enforced? Do you think that should lie with security or audit?

Interviewee: Audit

Interviewer: You think audit should police it?

Interviewee: I think so, uhm, because with all other controls patch management is a control to prevent a risk from patch management because it's a control and audit makes sure controls are there; they should do it.

Interviewer: Okay

14. Thank you for your time. Who else would you recommend I speak to?

Interviewee: Uhm, people outside our department would be xxxxxxx and xxxxxxx *(names removed to keep interviewees anonymous).*

Interviewer: You think that should be my next point of call?

Interviewee: Ya because they, I've dealt with them and from a business application side, probably speak to xxxxxxx.

Interviewer: xxxxxxx, thank you.

TRANSCRIPT NO 2 – ENTERPRISE ARCHITECTURE

1. What is your current job title and what previous job roles did you perform before you started your current job? (Previous companies and job roles)

Interviewee: Shoh, the current job title is easy, is Enterprise architect.

Interviewer: Okay

Interviewee: Okay, what did I do before this…uhm there's not in IT I haven't done.

Interviewer: Okay

Interviewee: So…I've been a network manager, I've been a programmer, I've been an analyst, I've been a…

Interviewer: You've been through it all.

Interviewee: Ya, I've been through the whole of development so there's nothing in development I haven't done. I've done DBA work, I've done, uhm, ya uhm, I mean I was fixing machines before you were born, uhm, I was fixing machines pre PC's [laughs] so, uhm yes, I've, uhm, together with a couple of other people manufactured boards from scratch [laughs].

Interviewer: So you've been, you're quite experienced obviously.

Interviewee: I've done a lot.

Interviewer: Okay

2. How familiar are you with the generic issues in patch management? (In general, not company specific)

Interviewee: Ya, well I mean, ya, I've obviously read quite a bit so I do understand a lot of it. I mean I've been working for this company for a long time so I'll clearly understand some of the patch management issues in this company…

Interviewer: Okay

Interviewee: …which are not necessarily that different to a lot of other companies, they, I mean, patch management there's a lot of commonalities across the board with patch management, uhm, but there are obviously some industries and some situations where patch management is a lot more difficult you know so especially some of the geographical things logistical type things, so how do you do patch management in a war zone for instance in Iraq? Very different to patch management here.

Interviewer: Okay [laughs]

3. **To what extent is patch management part of your job (What percentage of your time is spent on patch management functions)?**

Interviewee: No but not specifically, as an enterprise architect clearly I have to its part of my what I have to understand and part of what I consult on very often within the organisation but do I go and do patch management every day, no.

Interviewer: Okay

4. **How familiar are you with your company's patch management issues? (What are some of the current issues?)**

Interviewee: I'm much more familiar with patch management in terms of stores than for instance specifically with issues around head office…

Interviewer: Okay

Interviewee: …uhm although as I said there's a lot of commonalities amongst it and I understand some of the head office issues, uhm, I don't pretend to understand them all not in fine detail ya.

5. **Are you aware who the role players are in the patch management process? (Who is currently doing what?)**

Interviewee: Uh, yes

Interviewer: You know who they are?

Interviewee: Ya

6. **Who do you think should be added or removed from the patch management process?**

Interviewee: Uhm no it’s not a people problem. I mean patch management, patch management in stores in becoming better is becoming more under control uhm mainly because of the fact that they put SCCM in and they hold distribution mechanism[s] for software that has improved so that makes the concept of patch management and the way you patch much easier but you should understand stores have always been both more complex and simpler than head office; now I say that deliberately…

Interviewer: Okay

Interviewee: …because uhm simpler in the sense that stores in general have a common look and feel out there so there might be a generational difference you might XP out there, I think they’ve only just got rid of Windows 2000 so they might have Windows 2000 out there and Windows XP out there and Windows 7 out there…

Interviewer: …and then of course server 2008 uhm but within the XP environment within the windows 7 environment they all look the same so you don’t have variations of the theme within XP or variations of the theme within Windows 7 so the software sets that are on those various platforms are out there are all identical.

Interviewer: Okay

Interviewee: Okay so you don’t have to worry if you’ve tested for one windows 7 you’ve tested for all in theory anyway.

Interviewer: In theory, ja.

Interviewee: Uhm even the hardware is very very very similar in stores, if you go from one store to the next store there’s very little variance in the machines.

Interviewer: Okay

Interviewer: So there might be an older generation machine and a later generation but you know it’s in that way it’s simpler it’s more complex because it’s distributed far more so in the case of stores there’s 2500 stores uhm across a huge geographic area uhm so when you do something with a store whether it's a patch or a new piece of software whatever it is with a store you have to be very sure it worked before you start the distribution of it because if it gets to the 1651 store and things start to come unstuck; it's a little bit difficult to fix it because you already gone and [laughs] you know what I mean?

Interviewer: I'm with you, yes.

Interviewee: At head office you have a different scenario; here it's more complex. The variety of machines and the variety of systems that run on those machines are far greater than stores there's many more types of machines here there's many more types of software they run on those machines here but within a head-office context all the machines are in a fairly close geographic location and okay there are some regional offices as well but even then they're not too bad if things go wrong you can physically get to the machines and lay hands on them in stores you can't there's now ways that there are enough people to go to all of the stores and to fix them in a short time scale whereas in head office it might take even a couple of weeks to fix for instance a campus you can never the less do it in a couple of weeks in stores you couldn't do it in a couple of months.

Interviewer: Okay

Interviewee: Okay, very much more difficult so the two patch scenarios are very very different in the one case you've got to be extremely careful about before you actually start applying patches generically across the board…

Interviewer: Okay

Interviewee: …uhm in the other one you've got a lot more to test but…

Interviewer: In stores now?

Interviewee: No in head office.

Interviewer: Oh in head office.

Interviewee: There's a lot more testing to do okay because of the number of different machines and the number of different uhm pieces of software out there.

Interviewer: Okay

7. **What are the problems associated with managing patches from any external vendors? (What are the challenges when acquiring Microsoft or Adobe patches?)**

Interviewee: Uhm well for the…in general I don't really have problems acquiring patches, uhm, in general they're available you know some patches come out, Microsoft releases a monthly patch for instance a patch set and you apply those and occasionally you will have special patches which you have to apply that's only a result of a very direct fault so if something's gone wrong you inspect it you go and find what the symptoms are you go online and you put the symptoms in you get a thing that says that KB we'll fix that then and normally that's fine I mean we have one situation at the moment where dot net 4.5 has broken something in our environment, uhm, now that doesn't happen with everything in an environment in 4.5 but in a specific area it's broken it's an intermittent fault and it's not known to Microsoft so that's a really tricky one 'cause that's taken at that point in time three weeks of trying to track this particular thing and speaking to Microsoft and getting a patch out of them, we still don't have the patch because they still don't know what's going wrong yet…

Interviewer: Okay

Interviewee: So when you say are patches generally available, the common patches are and, and, if you took point of sale software for instance or service packs, they do three service packs a year and that's the way it comes out and because it's very rare that they at the end of a service pack cycle as they're implementing the first two couple of stores that they actually have fixes to that again 'cause their testing is so very thorough, they have to be very very thorough in their testing because you can't afford to break stores okay, and, but I do find that there's some anomalies in this whole patch thing. So for instance Microsoft patch is, they come down and they then you get things like Adobe which just spring in to life and just patch every now and again and uhm far less you are less able to control those for instances Adobe patches then you are Microsoft patches, uhm, I also find I find there for instance in the sort of Google in the Apple scenario this idea that they patch, I mean if you take a typical Google phone and Android phone, they're patching this thing three or four times a week, not at the OS level but at the application level, you've gotta ask yourself I mean I've gotta and there's not an awful lot of applications on this particular phone but I can guarantee that I've almost always behind on the updates, I dunno what they're doing I've got no clue, is this a big update or do they just want to know you're still there you don't know, so I think that's a less controlled environment.

In the Microsoft environment indeed even in the AIX because we've got an AIX upstairs and big oracle for instance uhm that's far far more stringent you know the patches are there they're well numbered, you know what they are, they've document what they fixed so you can go into any one of those things and they'll tell you what they fixed it's available to you. You can see the detail of what's happened on these type of devices Androids and that you've got no clue, uhm, in android itself you do and indeed in Microsoft the Windows on the phones when they put out a fix route you know exactly what they've done but it's all of the other software would be both telling you what they fixed or that you just get a…

Interviewer: Update

Interviewee: Ya, you get an update and that's it so you know you've got no clue.

Interviewer: Okay.

8. Are you aware of any test procedure for patches before they are deployed? (What is the current procedure? What do you think should the procedure be?)

Interviewee: Oh ya, in the stores what they're doing right now is they will, they are essentially they're testing a patch set, they'll test a patch set here in the lab initially…

Interviewer: Okay

Interviewee: They don't, they don't do extensive testing for an operating system patch set so they don't go and run every application through every process to make sure it works that's not what they do they do sort of basic applied patches you know a couple of a few basic tests because you'd drive yourself mad if you had to test you whole set every month you know you couldn't do that uhm so what they do is thy do the basic tests and then they take that software and they put it into a store…

Interviewer: Okay

Interviewee: …they'll patch a store and then they'll patch a set of stores let's say ten and then they'll take it slightly wider again they might do thirty or forty stores so they'll do internal, a small set uh one store just to make sure bigger set, bigger set and then everything and they do that over a period of about a month so there's normally about a week between each of those things.

Interviewer: Okay so there is a procedure in place for testing.

Interviewee: Ya

9. How are patches prioritised within the organisation? (Do you have any prioritisation policy in place or do you rely on that provided by software vendors?)

Interviewee: Both yes and no I mean at stores level we don't necessarily patch them there's their 72 hours we can't uhm but uhm we will apply all of their critical and important patches uhm within that month…

Interviewer: Okay

Interviewee: …that we do ya.

Interviewer: Critical and important

Interviewee: Ya

10. **How are patches acquired from different vendors? (How do you know what is available? What mechanism do you use to acquire these patches? How do you think this process scan be improved?) Is it strictly online downloads?**

Interviewee: Ya

Interviewer: That's the only mechanism you're aware of?

Interviewee: Uh not exclusively but mostly.

Interviewer: Mostly

Interviewee: Ya there are there are some patches that we apply for instance in the SAN environment the storage environment where the vendor comes on site and does our patches but those are quite technical uhm so we don't download something willy nilly and apply them they much much more studied and structured you know what I mean and the vendor comes on site, IBM in our case IBM comes on site and applies those patches so we don't just simply download something and patch it.

Interviewer: And tell me how do we know patches are available? How would we know to go and look for a patch now?

Interviewee: Well uhm it depends on what you're talking about uhm essentially with you know Microsoft it's pretty much patch and their patch Tuesdays, you know their patch Tuesdays is gonna happen of course we've talked about the other cases where you actually have a fault and you go look for it and then you know there's a patch for that so uhm and then on things like the SAN and that very typically it's done on a periodic basis that they will upgrade it…

Interviewer: Okay

Interviewee: …so uh unless again, unless there's a specific fault we're dealing with we don't upgrade outside of that cycle especially on the sand because the SAN I so critical to the organisation it's almost like the stores you can't break the SAN you know and if you break the SAN you're really in trouble and it's happened once or twice where they applied a patch and it's actually broken the SAN and then they've had to regress it uhm so those ones are done only at about six monthly intervals that kind of patch.

Interviewer: And tell me you say Microsoft Windows patch Tuesday, what if it's JAVA week?

Interviewee: Well that's why I say Adobe is a lot less controlled, you don't always know what's going on with Adobe you know and Adobe doesn't I suppose you can go and look for it, we don't typically as far as I know bother too much with Adobe as such uhm in the case of the stores we typically put Adobe out there we don't allow the automatic updates…

Interviewer: Okay

Interviewee: …because it would overwhelm us we just couldn't do it, so we typically don't patch Adobe that often maybe not often enough uhm but we don't so we don't typically we will control when Adobe gets patched through SCCM and it won't be even once a month.

Interviewer: Okay

Interviewee: It'll just be periodically.

11. Based on past experience, what incidents occurred as a direct result of failing to properly manage patches? (Malware infections? Security breaches? Integrity compromised? Confidentiality compromised?)

Interviewee: Uhm ya…I can tell you what went wrong when we applied patches I'll [unclear] framework 4.5…

Interviewer: Well that even too.

Interviewee: I mean uh that's a fairly recent one where it was applied uhm as per normal and in fact uhm it actually broke the Kerberos, now Kerberos is the authentication protocol that's used and in fact Kerberos broke but it didn't break entirely as I said intermittently Kerberos would fail to work correctly now Kerberos is quite a complex little protocol and it involved clock times and all sorts of things so it's hardly surprising that it broke uhm but break it did [laughs]. We have in the past been a bit suckered where we haven't applied patches properly and then when you do come to the point where you have to apply the patches you're so far behind that it becomes it doesn't become worth patching then it's a big and expensive exercise of re-installing operating system [audio interruption]. So I think when you look at the long term of an organisation especially in today's environment where you know everything is so well connected I think you lay yourself open to all sorts of abuse actually if you don't update.

Interviewer: Ya

Interviewee: Because a lot of their patches are security patches just you know blocking otherwise known hacking holes and stuff like so I guess if you look at an organisation like ours from a governance point of view you oblige to patch regularly you can't not patch otherwise you would be breaking a governance because you would make yourself open to abuse but also as I said you know we in the past found that er they didn't patch and then suddenly when you had to patch you couldn't, the past of patching of that much catch up…

Interviewer: Okay

Interviewee: …was just too great.

Interviewer: Okay

12. What components would you like to see go into a patch management policy? (What should be defined in the policy?)

Interviewee: Uhm…well there needs to be a set of principles about the patch management policy so you need to set up principles like…for what you, let me think this through properly… so you might say in principle we're going to apply patches one generation back so that's a fairly typical approach people take because they reckon I don't want to apply a patch until the rest of the world has applied it, if everybody did that we'd all be in trouble.

Interviewer: So that's the N-1 strategy?

Interviewee: Yes

Interviewer: Okay

Interviewee: So that might be a principle of what you, so you need to set up those principles first, you need to understand how you're operating in that environment and then in terms of the actual policy itself uhm you need to define how you're going to test and how deeply you're going to test okay you need to define your period of this test, how often, what your periods are…

Interviewer: Okay

Interviewee: …and ya patching patching patching ya it's the kind of thing you need to do. How you're gonna test, how often you're gonna test how you're actually going to deploy by the way is the other thing because there's several ways of deploying and you might have to have a deployment rule with different types of software so your own bespoke systems and whatsoever, this is how to deploy but Microsoft patches, this is how I deploy and Adobe that's how I deploy…

Interviewer: Why not one method for all?

Interviewee: I don't think it's, [yes don't forget my stuff] I don't think it's viable to do one method for all because they're different things if you know what I mean uhm so you know the period that you might do Microsoft patches might be more frequent then you do your own bespoke stuff okay because very often your own bespoke stuff the patching is more about extra functionality.

Interviewer: Okay

Interviewee: …whereas Microsoft stuff might be security patches so again I don't think there's a single rule but I think you can kind of categorise your rule set, in this category it's like this and like that in that category it's like that and that, that makes it much more manageable.

Interviewer: Okay

13. Who should develop the policy, implement the policy, and police the policy?

Interviewee: Uh, well, not a single person uhm typically uhm a patch policy that is developed between for instance operational department…

Interviewer: Okay

Interviewee: …uhm security department so in our case WSS who look after the servers they would be party to it.

Interviewer: Okay

Interviewee: Because they're looking after the environment after all. Uhm clearly your security department need to have a say and a hand in this and when it comes to your own software clearly the development area also needs to be part of that policy setting so it depends on what you're doing…

Interviewer: Okay

Interviewee: Ya

Interviewer: And tell me the implementation of this policy?

Interviewee: Hmm

Interviewer: Does it lie within operations only or should specific people be tasked with it or?

Interviewee: No, I think I don't I'm not a great believer in compartmentalising or silo-ing things that much because if you have a department that only does patching uhm they tend to become a little bit disconnected with the world and the rest of the world disconnected from them…

Interviewer: Okay

Interviewee: …so I would rather for instance put you can talk about distribution and actually applying of patches but you know the adoption and uhm testing of patches I believe gets done in the appropriate area so In terms of uhm our own stuff or packages we bought uhm that should be done in development areas in terms of this enterprise, so they should say we're going to apply this patch and we're going to test it, they might pass that on for instance in the case of stores to Richard's team and Richard's team actually deploy it and implement it, they execute the implementation in the stores.

Interviewer: Okay

Interviewee: Uhm, but I'm not a great believer in sort of having a whole patch department because it isolates them too much, they're not in touch with what's happening and you can't it's one of those things you can't simply willy nilly apply patches you have to understand what else is going on in the environment.

Interviewer: Ya

Interviewee: So it's no good me applying a patch to a piece of software that's about to be replaced, that would be you know a bit of a waste of time uhm and so you do it requires more, I just don't believe that it works well when you isolate it, or by silo-ing it.

Interviewer: Okay so we've got who develops it, who implements it and then who should police it. Who should make sure that this policy is being adhered to?

Interviewee: Uhm, well uhm, that's really the operational department so for instance if in terms of end-user uhm it is for instance uhm xxxxxxx's *(name omitted)* area that should be policing it okay and a lot of that policing is very automatic anyway, if we use the right tools it becomes policing by exception so it's the same way as we do for anti-virus, I mean, we don't run around and check every machine for anti-virus all the time, we simply have a system that says to us, these machines are two generations behind in their patch of anti-virus okay and that's the way you need to do it especially and in big organisations, the volume is just too great I mean I don't know how many terminals you've got at head office but it's probably in your head office and regions probably something like 7000, somewhere around there.

Interviewer: Yes

Interviewee: Uhm, Stores, it's another 6000 so you know you can't possibly manage that on an individual basis you do it by exception so you have a as they do so for instance in the stores they uses SCCM for the same purpose, they use it to collect version information…

Interviewer: …and uhm they monitor uhm monitor machines that are out of version for a particular aspect.

Interviewer: Okay

14. Thank you for your time. Who else would you recommend I speak to?

Interviewee: I dunno, who have you spoken to.

Interviewer: You can say the names, the ones already covered I'll ignore.

Interviewee: Uhm… xxxxxxx.

Interviewer: Okay

Interviewee: He understands stores far better than I do in terms of exactly what their process is in stores.

Interviewer: Okay

Interviewee: Uhm I would speak to xxxxxxx…

Interviewer: Okay

Interviewee: …'cause he can tell you a lot about patching for instance about the x environment for instance how they process that stuff, uhm, xxxxxxx in terms of the stores as well, uh, either xxxxxxx or you can even or even xxxxxxx.

Interviewer: Is it?

Interviewee: One of those two you probably already know from your own end-user perspective; best that you talk to them and ya probably those.

Interviewer: Thank you.

TRANSCRIPT NO 3 – OPERATIONS MANAGER

1. What is your current job title and what previous job roles did you perform before you started your current job? (Previous companies and job roles)

Interviewee: Currently I'm the head of Operations Manager for [company name removed] InfoTech and the SMC - the service management centre, uh responsible for all end-user computing, manage print services, AV support as well as VIP and MAC support, among other things.

Previous roles in the organisation is I've worked in the Windows server team as an IT Technical Specialist where I look after looked after the exchange enterprise vault, file servers and a couple of other application servers as well. Prior to that I was a service test team leader where we looked after a compliment of about twenty staff, uhm, and that was just day-to-day service desk help-desk related issues from a first and a second line point of view. Prior to that I worked in the USSS team well previously back then it was known as the LAN administration team where we did some server support from a server administration point of view so that was looking after components of exchange as well as looking after uhm file and print servers as well as restores and some other general admin uh prior to that I worked in the desktop support department within InfoTech; that was years ago and that's just normal desktop support related support, uhm, prior to that I worked at RCS group as a network administrator uh where I looked after their server environment; prior to that I was a desktop support first line uhm support analyst if you wanna call it that at that point in time for RCS group as well and prior to that it wasn't an IT related job so I don't think it counts right now.

Interviewer: Okay then.

2. How familiar are you with the generic issues in patch management? (In general, not company specific)

Interviewee: So generally patch management is always a, a complicated thing to actually manage especially in a big corporate such as ours at the moment, uhm, unfortunately we've got a lot of disparate applications, some of them Legacy some of them new and unfortunately once you do patch management, you know it has to be done in a very specific way or you could cause uh quite a bit of disruption within the environment, so let's say you patch something and Microsoft sort of plugs a specific security hole it might sort of break that specific application and then you'd have to roll back that patch uhm there are a lot of issues around Legacy operating systems like in our environment we still run XP and a large footprint a large footprint of our environment is windows XP and unfortunately patching those machines it could basically be very time consuming considering that it's not supported anymore by Microsoft, uh, as well as the fact that you know we haven't really had too many issues in that specific space considering that we have a lot of parameter protection from a security point of view. That's just Microsoft patching uh when it comes to application packaging there are even more complicated issues because you need a specific too which in the past didn't really exist to try and manage all your third party applications; thing like Adobe and you know those types of things so it does become a tiresome task if you don't have the correct tools in place especially from a legacy application because the way applications interface with the operating system as well as your uh let's say Office for instance like you know Planning, planning uses Excel and if you basically make a change to Excel, Planning might fall over and break and you know trying to roll back those patches and find out exactly where the issue is, is a very time consuming process.

3. To what extent is patch management part of your job (What percentage of your time is spent on patch management functions)?

Interviewee: So as it stands right now uh it is a big part of our job uhm it's we did it years ago uh we then ran a couple of projects where we, where we did a specific desktop optimisation project where we standardised our desktop environment by locking it down and doing specific things; at that point in time a call was made to uhm make sure that our image had service pack 3 installed on it and at that point it was the only cumulative service pack that was available to XP. That project ran for a while and we didn't actually do patch management for about three years or so, maybe it's a bit longer than that uhm now that we're moving to the Windows 7 space uh we're basically wanting to actively manage our desktops from a patch point of view and ultimately our desktop support configuration analysts, it's their responsibility to ensure that compliance levels are met.

Interviewer: Okay

4. How familiar are you with your company's patch management issues? (What are some of the current issues?)

Interviewee: So I'm fairly familiar because I've been here for while uhm again like I said in some of the previous questions or one of the previous questions the issues usually pertain to Legacy applications and some of the holes that a patch could resolve which could then negatively impact an application which unfortunately use that I wouldn't want call it vulnerability but use that loophole to actually work.

Interviewer: Okay

Interviewee: Uhm from an application or third party application management point of view again, the issues in that specific space is getting a third party utility to manage all of that so it's like I mentioned just now it's JAVA it's Adobe it's all these little things that people take into that they don't take into consideration that you need to basically patch, there are lots of tools out there and I forget the name of the tool that we are possibly gonna use in the future for managing those things uh but again that's a journey we still need to basically move forward on.

5. **Are you aware who the role players are in the patch management process? (Who is currently doing what?)**

Interviewee: Yes it's obviously from a…there's a matrix that's involved there and I can share that information with you so you have an understanding of who's all involved but from an end-user computing point of view obviously the desktop support or desktop configuration support analysts – the guys that actually look after our SCCM half of our images they're sort of the people that implement the patch management implement the patch management roll out uh the other role players who define policy and who define compliance is the security department and the security department would basically govern the rules of our organisation so they'll say we need to be 95% compliant we need to patch every patch Tuesday for instance or we can Patch every quarter dependent on what that policy is so us as [company name removed] InfoTech basically all the role players met from a management team point of view and we define that specific policy, we've got a RACI model around roles and responsibilities so like we are basically the owners and implementers of the patch management process where uhm the security team for instance they advise uh obviously there are other uh role players involved like the auditors department who actually manage or let's say they're the policemen uhm making sure compliance is met so that the security department is also sort of doing their job but security manages compliance and we act on what they define as our patch management policy is what our strategy is.

6. Who do you think should be added or removed from the patch management process?

Interviewee: I'd say…look you need to understand that because we weren't doing a lot of patch management we had to start from scratch and we were fortunate enough to have a lot of like in the security team for instance we had a lot of I don't really like to use the word- clever people but people with the technical knowledge and the experience from other organisation that started in our organisation uh who went through the turmoil's of patch management so ultimately they came in they gave us a lot of guidance around what we needed to do and they defined that specific policy and as it stands right now I think we have it spot on right now, my only concern is roles and responsibilities and when I say that it's yes I agree that end-user computing is responsible for rolling out those patches but like with everything else if your desktop functioning and working in my life it's functioning and working you know what I mean uhm my role as an operations manager is to make sure that the end-user is able to function and do what they do, patch management obviously adds a complexity to that environment; we need to understand that if something goes wrong it's not only my team that's gonna have to fix it; either we have to roll back but at the same time we need to engage with development teams and say okay this patch was applied, this is what it's done, you need to go back and re-develop or re-engineer your application so that it doesn't use that loophole or vulnerability that has been exposed uhm so to answer your question I think that we…the process and the people involved are right and uhm we're on a journey at the moment so ultimately going forward we're gonna learn a lot from the process which we're busy with at the moment.

7. What are the problems associated with managing patches from any external vendors? (What are the challenges when acquiring Microsoft or Adobe patches?)

Interviewee: So Microsoft is easy enough I mean we got the WSUS environment which integrates with our windows sorry our SCCM 2012 environment and ultimately the way it's configured at the moment is that, that WSUS service sucks down every patch that's required so it's in one source so it's not a sprawl around every work station going to collect patches from the internet and all our workstations are configured to point to the WSUS server so for Microsoft there's no problem because it's a slick process uhm the testing of that is another story but ultimately that's why we do it in phases in the form of pilot 1 pilot 2 and roll out to the rest of the environment when it comes to third party application packaging like sorry patching like uh Adobe and Java and things like that, that's where it becomes harsh because not harsh but it's a struggle, unfortunately you're very dependent on the internet for that unless you have a third party uhm patch management solution that like this one application you know I wish I had that name for you I'll share it with you a bit later where it acts as your repository for all patches similar to a WSUS server, you configure this thing and you say okay I wanna make sure that JAVA, I wanna make sure that Adobe, I wanna make sure that all those third party applications are there and ready to be patched at some point in time and we can point all our work station to that service to actually patch it so there are tools out there, they're costly uh and they're sort of licensed over x amount of years so you pay it's almost like a subscription cost that you pay for that service uh so you pay for it more than anything whereas the WSUS service is usually included in your Microsoft agreements.

Interviewer: Okay then just to confirm on the testing [leads into next question].

8. Are you aware of any test procedure for patches before they are deployed? (What is the current procedure? What do you think should the procedure be?)

Interviewer: Ultimately we'll follow exactly the same process from a non-Microsoft based patching versus a Microsoft based patch type of process so ultimately the rule of thumb is to first basically deploy to a small group of users and we use InfoTech users as our guinea pigs f you want to call it that just to confirm that the process is actually sound and that it works so we'll deploy to x amount of people make sure you know that all the rule sets we have in place around alerting the user that your machine might reboot in 60minutes or give them the ability to actually apply the patches within a time-period is there so uhm ultimately what was the question again? [laughs].

Interviewer: Are you aware of any test procedures?

Interviewee: Test procedures ya. So that first patch is just to test the process and make sure the application is installed, not the application, the patch is installed and that there aren't any problems in the deployment process once that's done we move on to pilot 2 if you want to call it that or phase 2, uh, where we choose specific users in our business; preferably people that use a large variety of applications in that business unit. Now we're unique to a point because we've got different trading divisions in our environment so you know it's not like other companies where it's Truworths for instance and it's just Truworths you know what I mean, we the Foschini group and we've got all these brands and all of these brands have different applications sets used I mean you know you work in the environment uhm and unfortunately we can't do a one size fits all so we basically ear-mark five pilot users in each division and we make sure that those users are using a compliment of the application for that specific division, we deploy to them, we give it a week or so just to confirm that all the applications sets work we then elicit feedback from them and find out has everything you know how was the process uhm did anything break uh we also have mechanisms I place with a service test to say these are the users that are basically on this pilot uhm what we're doing is we're trying to pull a report daily to see how many calls they've logged in the pilot process.

If that pilot process is successful we'll then say okay you know what we've got business sign off that there weren't any issues uh we will then basically run our pilot per division not the pilot but the balance of the environment per division so we'll tackle Foschini, we'll tackle Markham, we'll tackle you know whichever other division is in place.

9. **How are patches prioritised within the organisation? (Do you have any prioritisation policy in place or do you rely on that provided by software vendors?)**

Interviewee: So like you know there's Patch Wednesday sorry Patch Tuesday uh ultimately Microsoft recommends that you patch every Tuesday; that's not practical in a corporate such as ours, you'd probably need a dedicated patch management team and you know here we sort of use our resources for everything possible and I see you laughing but anyways so we've made a call that we will only patch every quarter. Dependent on your organisation you can, like let's use the store's environment for instance because I'm speaking specifically for head office: in a store's environment it's active retail and a couple of office applications so it's a very easy test pilot phase that you can go though, in the head-office environment and they could possibly patch every Tuesday if they wanted to but I think they also made a call to patch every quarter considering the large footprint but in our environment there's way too many applications that could be impacted on for us to patch every Tuesday so I work on the rule of thumb that Microsoft can make mistakes so you could possible go and patch something and then there's a fault on that specific patch 'cause it's happened in the past and I've experienced it where we patched something; it's broken IE to hell and gone uhm and then Microsoft will release something the next day saying okay we've had a couple of complaints too bad so sad please install this specific patch uh it's a little bit late at that point in time so I work on an if you want to call it a N-1 type of a scenario if I were running a smaller organisation I would always say and I could patch every Tuesday I would say I would always patch last week's patches this week so we had a clear indication around uhm our anybody else that might have experienced problems that actually patch every Tuesday.

In the Foschini group environment we decided to go every quarter just because it's quite a big workload for one specific team to look at.

Interviewer: Okay.

10. How are patches acquired from different vendors? (How do you know what is available? What mechanism do you use to acquire these patches? How do you think this process scan be improved?)

Interviewee: It's WSUS and downloads, mainly. I mean look right now we do ad hoc patch management in our let's Adobe reader uhm and our DCS guys they'll basically pick up a bug or there's an alert that says you need to basically patch Adobe, we'll then download the patch and deploy it via SCCM, so it's mainly from a download point of view. We do have other applications uhm where there are vendors involved and the vendors will recommend that you upgrade in x amount of time but that rarely happens.

11. Based on past experience, what incidents occurred as a direct result of failing to properly manage patches? (Malware infections? Security breaches? Integrity compromised? Confidentiality compromised?)

Interviewee: Let's use, we had an issue once upon a time…okay let me start here: InfoTech is quite secure from a parameter point of view now the security department protects our parameter quite well so nothing can really get into our environment from an outside point of view well I suppose if someone tried hard enough and they were clever enough they could get in but you know we haven't had those types of issues in the past. From an internal point of view because you obviously have USB sticks and people with all their viruses and and and and we basically secure our desktop using our anti-virus and we trust that semantic is actually gonna release stats and that all our compliance levels on our in-points are up to date so that usually picks up those issues [audio interruption: phone rang] so in my view we did have one issue and I don't know if you remember the Conficker virus?

Interviewer: Yes I've heard of it.

Interviewee: Ya, so ultimately because we didn't patch at that point in time and with our XP world we were hit with that specific virus because it played on that loophole that vulnerability within the windows operating system so it caused a whole lot of issues in our head office environment and I forget now but I remember still spending a whole lot of time at work trying to get that issue resolved but you know we spent an entire weekend uhm basically having to deploy the patch that actually patches that vulnerability and that was a direct result of not patching and that wasn't coming from the outside it was somebody very naively coming in with a USB stick because they did something at home and it basically spawned this virus in our environment so that is the one example in the years of experience that I've had at Foschini that was a direct result of not basically patching so it is an important thing uhm and unfortunately you can't catch it a lot but you know straight after that I know they endeavouring patching quite a bit at that point in time.

12. What components would you like to see go into a Patch Management policy? (What should be defined in the policy?)

Interviewee: So I mean like I said we've got a patch management policy which I'll share with you, look it's I'm not saying you can use all the data to plagiarise it in any way but it will give you an idea around what patch management policy defines so ultimately first you need a Racy model to basically understand the roles and responsibilities for the different teams because it's for a couple a team's 'cause it's not only the desktops, there's the server team uhm then there's also the security department and there's the management you know Esco where they basically have, they need to have some insight around what we are patching so that they can report back to the board from a compliance point level of view. So firstly a RACI model is required then we basically we need to define what the process of patching is so that would a little Visio workflow diagram that says pilot 1 pilot 2 to if this happens we can continue, if this happens roll back you know, you know that whole Visio type of process for the document. Then also just defined, when I say roles and responsibilities in that RACI model there's usually a table but define in words who's responsible for what from a policy point of view, uh, to end that off with you need to also basically make sure that there are a couple of disclaimers involved around what you define as your patch management policy because you can I mean you know there's important, there's critical security, there's not so important and that's not a technical term but you should know that there are a few categories from a patch management point of view, we've decided to go critical and security patches only inn our environment 'cause the critical patches usually take care of the hard core stuff the important and the nice to haves are okay but they usually take care of a silly thing, it's a bug in Microsoft sorry it's a bug in office for instance you know what I mean and they don't really cause too much of a threat to the environment, they can possibly cause problems for the desktop support team because 'oh no this is not working' but ya so critical uhm and security are the most important ones, in that policy you need to define what it is that you want to patch but I'll give you some insight to the documents that you can understand what it looks like and you can maybe just use it as a, I'm not saying a template but something that you can use as a point of reference.

13. Who should develop the policy, implement the policy, and police the policy?

Interviewee: It is a combined effort with the operations teams involved so and the server team obviously, there's the desktop support team and, uhm, the security department as well so those are the people that need to basically define that policy so. Security will draft the policy and will then pull everybody into a room and say are you happy with X, Y and Z, we'll give our feedback they'll then amend that specific policy, they'll send it back to us we will say no we don't like that, send it back and that's a bit of a give and take so it takes a bit of time to actually develop a policy but it's to make sure that we meet compliance first and fore-most uh that we aren't vulnerable to an attack uh but also to make sure that the operational overhead isn't too taxing on your team dependent on the staff members that you have.

Interviewer: And then who should be responsible for the implementation of this policy?

Interviewee: So when you say implementation of the policy I'm assuming…

Interviewer: Physically doing the work.

Interviewee: Following the policy?

Interviewer: Yes

Interviewee: [coughs] Again your policy will have your RACI model so it'll have roles and responsibilities so it's usually the technical staff, technical operational staff would be the implementers of the policy so they get told that you need to apply all critical and security patches they will then use whatever tools that they have at their disposal to implement.

Interviewer: And then who should be responsible to police this policy?

Interviewee: The security department.

Interviewer: So, okay so basically…

Interviewee: They should basically be the, 'cause they're the ones that govern our security compliance level uhm and obviously there's somebody that needs to manage them and that's the auditors. I don't know if every organisation has an internal audit department but ultimately our internal audit department will then basically make sure that the policies that are in place are ultimately in line with what the requirement is from an industry standard point of view.

Interviewer: Okay.

14. Thank you for your time. Who else would you recommend I speak to?

Interviewee: Maybe and I don't know if you've had a conversation with them but xxxxxxx who's actually the person who's designed and implemented our actual WSUS and SCCM environment, so if he has some time maybe interview him to give you an understanding of the technical detail around how you deploy and what you deploy and you know the pro's and the cons of deploying. Look, nobody knows what's gonna go wrong from a patch management implementation point of view but I would say if you want a full understanding it's nice to get the people on the ground that's actually well I wouldn't say on the ground, somebody that understands the design and architecture of our environment and how that implementation works because xxxxxxx's responsible right now for designing that solution, making sure that the policy is, that we meet what the policy dictates uh documenting that process, so it's a process it's not a policy that can be handed over to operations that the operational teams are able to actually deploy and use that technology, whatever technology is at their disposal. So xxxxxxx's a good uh a good person to actually speak to from a patch management process point of view so pilot 1 pilot 2 he can give you some detail around why we do it that way and why we basically do it.

Interviewer: Okay thank you, xxxxxxx.

Interviewee: Cool, not a problem.

APPENDIX B: EXAMPLE OF CODING TECHNIQUE APPLIED (LATENT CODING)

What components would you like to see go into a patch management policy?

Interviewee: I think you know a key component is what do you want to patch I think you know uhm if for instance if you run an exchange server you need to decide uhm and it needs to be granular as granular as possible so you need to decide you know I'm only gonna patch windows so I'm gonna treat applications as a separate uh s you don't have that all-encompassing patch strategy that says you know this covers all technology and all applications that I run because applications are very different to the base OS' that you run so I think key components that needs to feed into your strategy is: what do you want to address, what is the risk that you, that's why we decided on security patches critical patches kind of thing because we didn't want to you know patch uhm the windows installer and patch the windows firewall service and patch you know because those aren't even functionalities that we use kind of thing so we didn't wanna go in with a strategy and like I said any change in to your environment has a sense of risk attached to it kind of thing so we wanted to say industry best practice what should we focus on and I think that's key, is to say what is the scope of my patching gonna involve and have that for your core OS for one like we've got in both AIX and Windows and then also it speaks to your server patching as well where your firmware comes into play of your actual hardware devices kind of thing so I think the key component is to say how big is the scope of my strategy and what I'm going to be patching and then decide what is your appetite for risk in the sense of am I gonna always go the latest and greatest as soon as it gets released I'm gonna apply it or are you going into this whole what is the period that you're gonna apply your patches and we've decided on twice yearly so every six months uhm and to me that's kind of key but so you can have an overall strategy uhm but then with very key metrics inside and say this is the scope, this is what I'm gonna focus on and this is the time period on how I'm gonna address that.

Interviewer: Okay.

APPENDIX C: UNSTRUCTURED IDENTIFIED THEMES

- Patch management forms a small part of current job function
- Participants find it difficult to quantify how much time is spend on patch management activities
- The security department is unaware of the challenges faced in the organisation
- All participants have an idea of who the role players are and what the process is, but no one has a complete view of the entire process
- Misconceptions exist regarding who does what, who is involved, and who should be involved
- Roles and responsibilities are not clearly defined
- Change management and application owners may need to be added to the patch management process
- Isolated patch management activities are not being monitored and documented
- There is no defined testing strategy
- Testing is done on an ad-hoc basis
- The test environment is out of sync with production environment
- Release patches are often not well documented in terms of version number, what they fix, and release date
- Vendors have inconsistent behaviour when I come to releasing patches
- It is a challenge to maintain a current software register
- Legacy systems pose a variety of challenges to patch
- Resource constraints are a limiting factor
- Third party tools are expensive and patch activities is time-consuming
- Patch management process is complex and adds to the complexity of the environment
- Importance of patch management is not well understood
- Communication between role players is a problem
- Systems that have not been patched in a long period of time is more risky to patch on the initial attempt
- Inventory reports are skewed for a number of reasons
- The risk of patching has not been identified
- Communication between roles players is a challenge
- The security department is not as involved as they should be
- There is a lack of consistent patch activities
- There are no defined maintenance windows for patch deployments

- There is no prioritisation mechanism or response procedure in place
- The organisation is very risk averse
- Complete automation of patches poses challenges for testing
- The organisation is often forced to align with the vendor's patching strategy
- Too much red-tape exists in order to approve patch deployments
- None of technical teams in the process have an understanding of the challenges faced by technical staff
- Patch management activities are not consistent
- Testing challenges; there is no real test environment, and the testing is a resource intensive process
- The informal testing procedure is that patches are testing on a virtual machine for a while, and then released into production in a phased approach
- Lack of proper communication mechanisms often lead to wrong decisions being made about patch management related issues
- The patch management policy should be developed by the security department, implemented by operational teams and policy by a joined effort between internal audit and the information security department
- Software vendors do not provide patches for older versions of software that is no longer supported
- Microsoft has a smooth process when it comes to patches, but most other vendors do not
- Only security patches and critical patches are installed
- The frequency of patch deployments has not been defined
- More incidents were reported as a result of applying patches, than what was reported as a result of not applying patches
- The organisation suffered extensive losses from the Conficker worm outbreak
- Possible components that should be covered in the patch management policy include:
    - roles and responsibilities
    - information awareness
    - identification od stakeholders
    - categories of patching
    - frequency of patch deployments
    - testing strategy
    - patching strategy
    - set of principles

    - deployment mechanism
    - categories of patching plus rules sets per category
    - the process defined in words
    - the importance of patching explained
    - compliance requirements
    - the testing strategy
- The policy should ideally be signed by senior management
- The policy should reduce red-tape and eliminate certain current challenges
- The majority of all security related incidents are as a result of vulnerabilities being exploited
- A patch management policy should play a major role in information security in an organisation
- Patch management is often not an easy task
- The time between discovering vulnerability and patching is often too long
- After vulnerabilities are disclosed, attacks first increase before they decrease
- The majority of the user community do not keep systems up to date
- There is large number of systems that cannot be restarted due to availability requirements
- Companies can benefit from having a patch management policy in place
- Challenges in patch management can lead to compromises in enterprise data security
- Patch management is one of the main components of IT management
- Patching is done to fix security and critical vulnerabilities
- Vulnerability management and patch management should be economically controlled
- There is a lack of means to directly measure the effectiveness of a security mechanism
- Patch related information can be obtained from a variety of sources
- More than 50% of vulnerabilities that exist in the world have patches released
- Deciding what patches to install is a challenge
- The most notable exploit of 2014 was the Heartbleed exploit
- The patch management process is often flawed
- Security breaches can have significant impact on economies
- Vendors typically do not support older versions of software
- The mapping between techniques, vulnerabilities, address, and administration are often skewed
- Organisations have constraint resources

- Security breaches are a regular occurrence
- There is significant risk in installing patches that have not been tested
- Testing is resource intensive
- Installing patches can break working systems
- Certain systems cannot be started or shutdown, making them a challenge to patch
- Infrastructure can introduce certain challenges, such as bandwidth constraints
- Timing, prioritisation and testing are often in direct conflict
- All platforms should be addressed in the patch management policy, such as mobile device, server, and workstations network nodes
- Third party tools can help with software maintenance
- Patches are a time consuming process; insufficient time is listed as one of the main reason why patches are simply not installed
- Configuration systems will add complexity to the environment
- Testing challenging: too little testing and unexpected results may occur after deployment, too much testing and the vulnerability maybe exploited before the patch is deployed
- Patch management is dependent on having current inventory information available
- Conficker worm affected users around the world, after a patch had been released
- Discovering vulnerabilities is a challenge
- The mechanism used to deploy patches may be a challenge
- There are very little defences against zero-day attacks
- A large number of vulnerabilities continue to be exploited after a patch has been released
- It is often the perception that patches contain vulnerabilities
- User apathy and inefficient security controls contribute to lack of patching
- No single system supports all software
- Complete automation of patch deployment is challenging in terms of testing
- Complete automation implies complete trust
- Different software platforms require different approaches
- Solutions may have different limitations; it often does not scale well
- Response mechanism should form part of the policy
- The way a vulnerability has been disclosed can have an effect on whether a vulnerability is exploited or not
- Some patches introduce new vulnerabilities
- If users perceive a patch to be complex and risky, they may choose to skip it all together

- Role players include system administrators, vendors and government; patch management is a sub-set of release management
- There are no standardised concepts; the same concepts may have different meaning to different stake holders
- ITIL release management roles are not well defined
- Lack of information of communication procedures
- Lack of metrics for patch management
- Tendency to supersede maintenance issues for production releases
- System administrators do not have strong enough support from management
- System compromises could often have been avoided if patches were up to date
- Allowing users to manually install patches is not a good idea
- Consideration should be given to the deployment method, as there are several searches with their own benefits and side effects
- Technical challenges, disk encryption, etc.
- Homogeneous host architecture makes it more challenging to manage patches
- Security needs should be balanced with that of usability and availability requirements
- If a system is too secure, users will go out of their way not to use it
- Finding maintenance time windows are difficult
- Change management is vital as process stretches over various departments
- Testing is important
- Code advance has attempted to reduce vulnerabilities in software
- There is no central location where patch information can be found
- Patches should be categorised as far possible
- Hot patching may be a possible solution for live updating of systems when there is legislative pressure; vendors respond quicker
- Public disclosure can increase the risk of security attacks
- Legislation can be a useful tool

APPENDIX D: STRUCTURED IDENTIFIED THEMES

MAIN THEMES

- Patch management activities are not consistent
- The security department is not as involved and updated as they should be
- No party is looking at the complete picture; all activities are occurring in silos
- Testing is a major concern in several aspects
- Vendors have inconsistent behaviour when it comes to patch management and the organisation is forced to align with the vendor
- Processes are not always well documented
- Legacy systems pose a variety of challenges for the organisation
- Third party tools are costly and require skilled employees to manage the tools
- Communication challenges have a direct effect on challenges experience relating to patch management
- The lack of current and up to date inventory reports poses a challenge
- Patch related risks have not been identified
- There are no set maintenance windows in which to do the patches, making it difficult to prioritise patches
- Roles and responsibilities are not clearly defined
- Patch policy should be signed off by a person in authority; that way it can reduce red tape
- Patch management adds complexity to the environment

SUB-THEMES

- In general, patch management forms a small part of participants' current job functions
- Participants find it difficult to quantify how much time is spend on patch management activities
- All participants have an idea of who the role players are, but no one is looking at the complete picture
- Change management and application owners may need to be added to the patch management process
- Resource constraints are a limiting factor
- The Importance of patch management is not well-understood
- Systems that have not been patched in a long period of time are more risky to patch
- There is not prioritisation mechanism or response procedure in place

- The organisation is very risk averse
- The organisation is often forced to align with the vendors’ patching strategy
- Patch management activities are not consistent
- The patch management policy should be developed by the security department, implemented by operational teams, and policed by a joined effort between internal audit and the information security department
- Software vendors do not provide patches for older versions of software that is no longer supported
- Microsoft has a smooth process when it comes to patches, but most other vendors do not
- Only security patches and critical patches are installed
- The frequency of patch deployments has not been defined
- More incidents were reported as a result of applying patches than what was reported as a result of not applying patches
- The organisation suffered extensive losses from the Conficker worm outbreak
- Possible components that should be covered in the patch management policy include:
    - roles and responsibilities
    - information awareness
    - identification of stakeholders
    - categories of patching
    - frequency of patch deployments
    - the testing strategy
    - patching strategy
    - set of principles
    - deployment mechanism
    - categories of patching plus rules sets per category
    - the process defined in words
    - the importance of patching explained
    - compliance requirements
    - the testing strategy

THEMES AND IDEAS COLLECTED FROM SECONDARY DATA (LITERATURE REVIEW)

MAIN THEMES

- The majority of the users simply does not apply patches for a variety of reasons
- High availability system poses a challenge to patch as they cannot be rebooted or shut down
- Patch management is directly linked to enterprise data security
- There is a lack of metrics to measure the effectiveness of security controls
- Vendors typically so not support older versions of software
- Testing plays a crucial function within patch management
- Patch management addresses complexity to the environment
- Organisation has constrained resources
- Patch management relies heavily on current inventory information
- System administrators often do not have strong enough support from management
- There is no single solution that supports all systems
- Different software platforms require different approaches toward patch management
- Lack of communication procedure is considered a challenge
- There is no centralised source of patch information
- It is important for the policy to be signed off by senior management
- Legislation can be a useful tool within the patch management process
- All risks related to patch management should be identified and classified
- There is a lack of information documentation and communication procedures

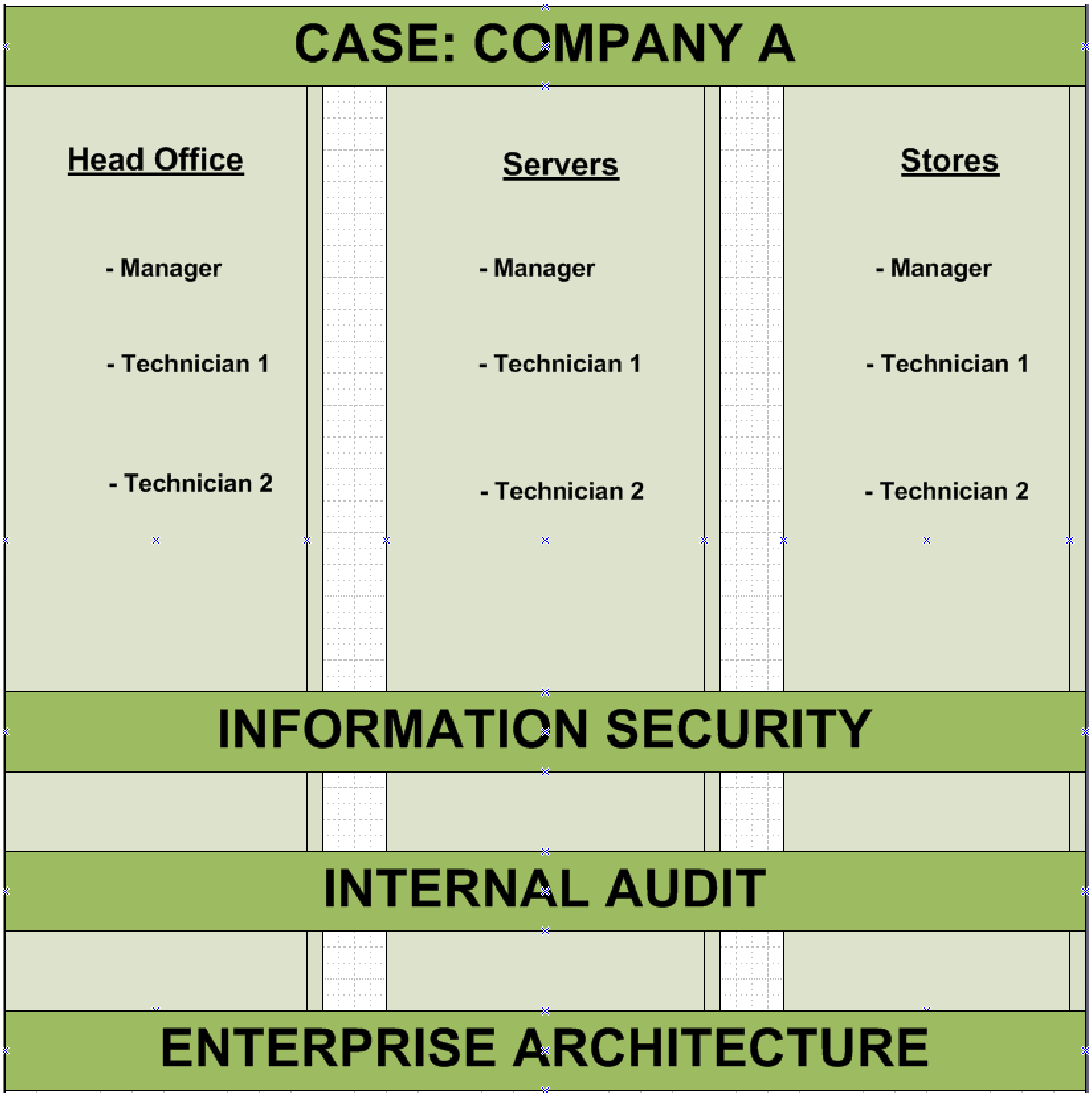
SUB-THEMES

- The majority of all security related incidents are a result of vulnerabilities being exploited
- Patch management is a complicated task
- The patch management policy should play critical role in the overall security of an organisation
- The time between discovering a patch and deploying a patch is often too long
- There is little defence against zero day attacks
- An organisation should find the balance between security, usability and availability
- No defined maintenance windows make patch deployments more challenging
- Documentation is vital throughout the process

- A baseline policy combines with a risk analysis can be a good place to start for an organisation wanting to adopt a patch management policy
- Patches should be categorised as far as possible
- The time between discovering vulnerabilities and patching is often too long
- After vulnerabilities are disclosed, attacks first increase before they decrease
- Patch management is one of the main components of IT management
- Patching is mainly done to fix security and critical vulnerabilities
- Vulnerability management and patch management should be economically controlled
- More than half of all vulnerabilities that exist in the world have patches released
- The most notable exploit of 2014 was the Heartbleed exploit
- The patch management process is often flawed
- Security breaches can have significant impact on economies
- Security breaches are a regular occurrence
- Installing patches can break working systems
- Infrastructure can introduce certain challenges, such as bandwidth constraints
- Timing, prioritisation and testing are often in direct conflict with one another
- Third party tool can help with software maintenance
- Patch deployment is a time-consuming process
- Insufficient time is listed as one of the main reason why patches are simply not installed
- Configuration systems will add complexity to the environment
- Testing poses challenging; too little testing could result in unexpected behaviour after patches have been deployed. Too much testing and the vulnerabilities may be exploited before the patch is deployed
- Patch management is dependent on having current inventory information available
- Conficker worm affected users around the world, after a patch had been released months before
- A large number of vulnerabilities continue to be exploited after a patch has been released
- It is often the perception that patches contain vulnerabilities
- User apathy and inefficient security controls contribute to a lack of patching
- No single solution supports all software systems; solutions may have different limitations
- A response mechanism should form part of the patch management policy
- The way a vulnerability is disclosed can have an effect on whether vulnerabilities are exploited or not

- Some patches introduce new vulnerabilities
- If users perceive a patch to be complex and risky, they may choose to skip the patch
- Role players include system administrators, vendors, and government
- Concepts are often not standardised; the same concepts may have different meaning to different stake holders
- System compromises could in many cases have been avoided if system patches were up to date
- Allowing users to manually install patches is not a good idea
- Consideration should be given to the deployment method, as there are several methods, each with their own advantages and disadvantages
- If a system is too secure, users will go out of their way not to use it
- Change management is vital as the patch management process stretches over various departments
- Software programming advances has attempted to reduce vulnerabilities in new software
- Hot patching may be a possible solution for the live online patching of systems
- Public disclosure can increase the risk of security attacks
- Parameter security may be seen as an alternative to patching
- Patch management can leverage off existing frameworks
- Security patches may cause system failures
- Patching should be a shared responsibility between the software vendor, private organisations, and government
- Release management and patch management are tool-oriented, and tools should be efficient
- The environment should be standardised as far possible
- Organisations should be able to distinguish between vulnerabilities that affect them versus vulnerabilities that do not affect them
- A patch management policy can lead to cost savings if resources are properly coordinated
- Security controls should be implemented properly otherwise they can do more harm than good
- Keeping software up to date with the latest versions reduce the need for patch deployments
- Infrastructural limitations such as available bandwidth should be considered when designing a patch management policy
- Maintenance windows should allow enough time for a rollback procedure as well

APPENDIX E: REPRESENTATION OF RESEARCH SAMPLE



APPENDIX F: RESEARCH INSTRUMENT: INTERVIEW QUESTIONS

1. What is your current job title and what previous job roles did you perform before you started your current job? (Previous companies and job roles)

2. How familiar are you with the generic issues in patch management? (In general, not company specific)

3. To what extent is patch management part of your job? (What percentage of your time is spent on patch management functions?)

4. How familiar are you with your company's patch management issues? (What are some of the current issues?)

5. Are you aware who the role players are in the patch management process? (Who is currently doing what?)

6. Who do you think should be added or removed from the patch management process?

7. What are the problems associated with managing patches from any external vendors? (What are the challenges when acquiring Microsoft or Adobe patches?)

8. Are you aware of any test procedure for patches before they are deployed? (What is the current procedure? What do you think should the procedure be?)

9. How are patches prioritised within the organisation? (Do you have any prioritisation policy in place or do you rely on that provided by software vendors?)

10. How are patches acquired from different vendors? (How do you know what is available? What mechanism do you use to acquire these patches? How do you think this process scan be improved?)

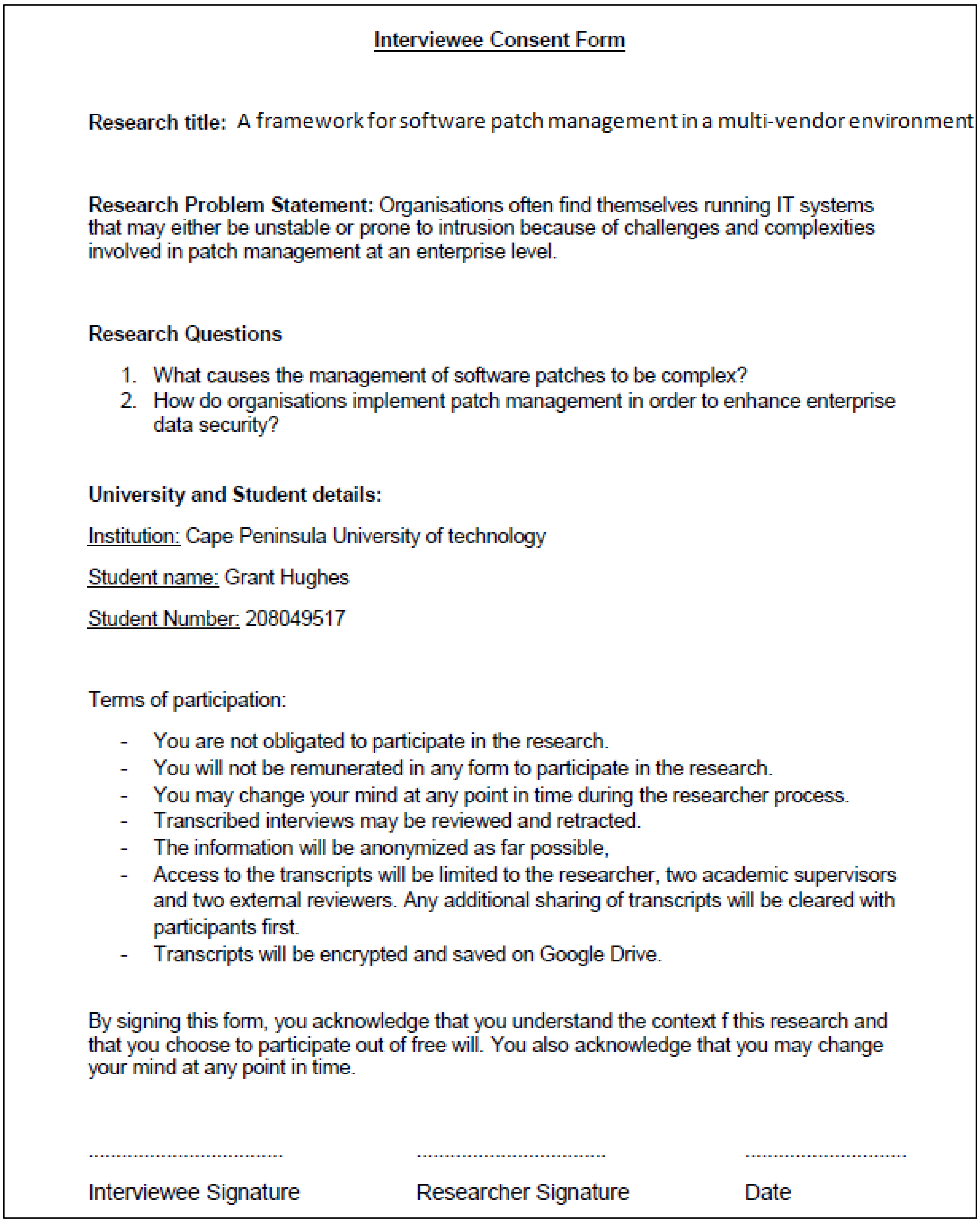
11. Based on past experience, what incidents occurred as a direct result of failing to properly manage patches? (Malware infections? Security breaches? Integrity compromised? Confidentiality compromised?)

12. What components would you like to see go into a patch management policy? (What should be defined in the policy?)

13. Who should develop the policy, implement the policy, and police the policy?

14. Thank you for your time. Who else would you recommend I speak to?

APPENDIX G: CONSENT FORM: PARTICIPATION IN CASE STUDY RESEARCH



Interviewee Consent Form

Research title: A framework for software patch management in a multi-vendor environment

Research Problem Statement: Organisations often find themselves running IT systems that may either be unstable or prone to intrusion because of challenges and complexities involved in patch management at an enterprise level.

Research Questions

1. What causes the management of software patches to be complex?
2. How do organisations implement patch management in order to enhance enterprise data security?

University and Student details:

Institution: Cape Peninsula University of technology

Student name: Grant Hughes

Student Number: 208049517

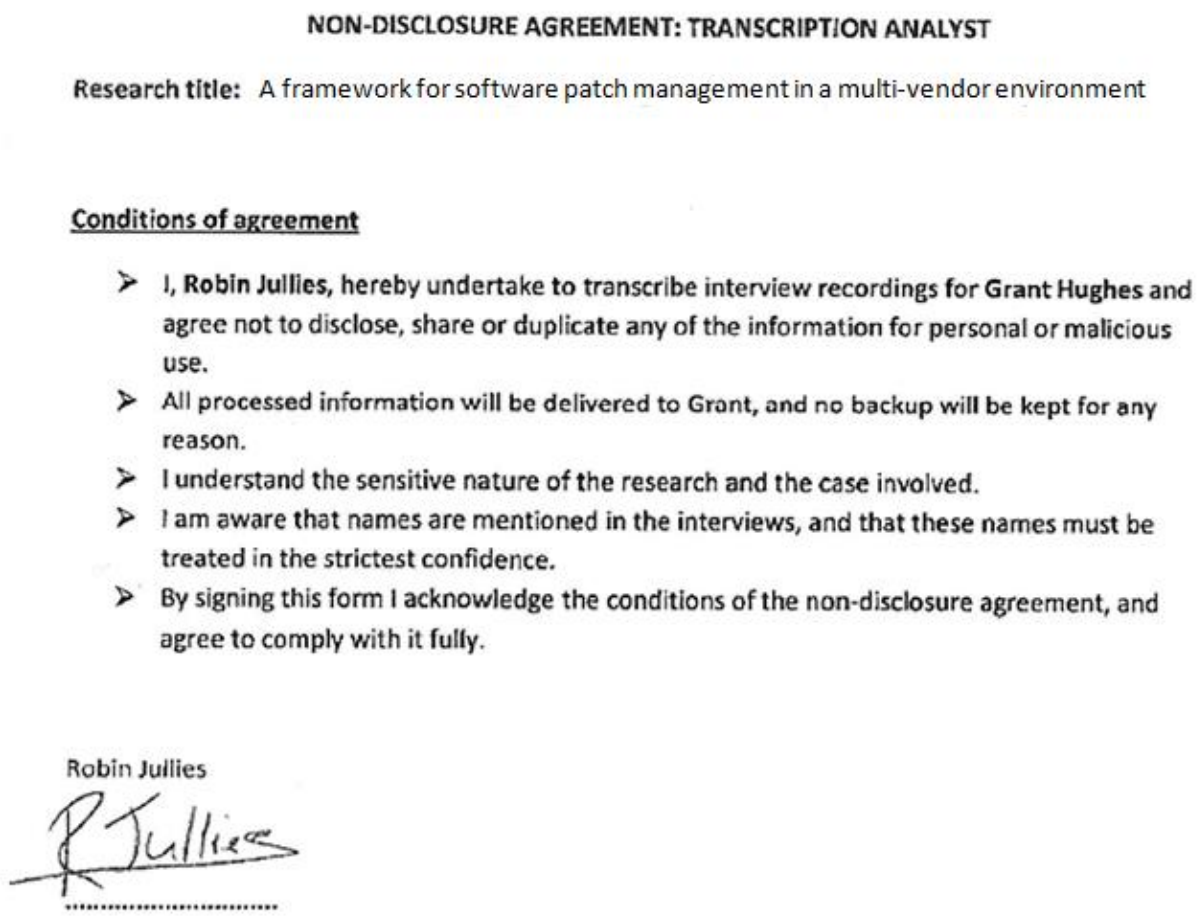
Terms of participation:

- You are not obligated to participate in the research.
- You will not be remunerated in any form to participate in the research.
- You may change your mind at any point in time during the researcher process.
- Transcribed interviews may be reviewed and retracted.
- The information will be anonymized as far possible,
- Access to the transcripts will be limited to the researcher, two academic supervisors and two external reviewers. Any additional sharing of transcripts will be cleared with participants first.
- Transcripts will be encrypted and saved on Google Drive.

By signing this form, you acknowledge that you understand the context f this research and that you choose to participate out of free will. You also acknowledge that you may change your mind at any point in time.

....	....	....
Interviewee Signature	Researcher Signature	Date

APPENDIX H: NON-DISCLOSURE AGREEMENT: TRANSCRIPTION ANALYST



NON-DISCLOSURE AGREEMENT: TRANSCRIPTION ANALYST

Research title: A framework for software patch management in a multi-vendor environment

Conditions of agreement

- I, **Robin Jullies**, hereby undertake to transcribe interview recordings for **Grant Hughes** and agree not to disclose, share or duplicate any of the information for personal or malicious use.
- All processed information will be delivered to Grant, and no backup will be kept for any reason.
- I understand the sensitive nature of the research and the case involved.
- I am aware that names are mentioned in the interviews, and that these names must be treated in the strictest confidence.
- By signing this form I acknowledge the conditions of the non-disclosure agreement, and agree to comply with it fully.

Robin Jullies

[illegible]

APPENDIX I: INTERVIEWEE QUOTES TO INTERVIEWEE AND TRANSCRIPT MAPPING

The table below presents all the direct quotes that was made by interviewees and extracted from the interview transcripts. All interview transcripts are available upon request. The quotes are presented in ascending order from interviewee 1 to interviewee 13. A mapping of the interviewee's identifier to the actual interviewee's name is available on request, but is being intentionally withheld in order to maintain anonymity of interviewees.

Quotes made by interviewees that were extracted from interview transcripts	**Interviewee (I) who made the statement**	**Transcript page number (separate documents - available on request)**
"A complicated thing to actually manage especially in a big corporate such as ours at the moment, uhm, unfortunately we've got a lot of disparate applications, some of them Legacy some"	I1	1
"My only concern is roles and responsibilities"	I1	4
"In that RACI model there's usually a table, but define in words who's responsible for what from a policy point"	I1	8
"Patch management obviously adds a complexity to that environment; we need to understand that if something goes wrong it's not only my team that's gonna have to fix it; either we have to roll back but at the same time we need to engage with development teams and say okay this patch was applied, this is what it's done, you need to go back and re-develop or re-engineer your application so that it doesn't use that loophole or vulnerability"	I1	4
"First patch is just to test the process and make sure the application is installed, not the application, the patch is installed and that there aren't any problems in the deployment process once that's done we move on to pilot 2 if you want to call it that or phase 2, uh, where we choose specific users in our business; preferably people that use a large variety of applications in that business unit"	I1	5

Quotes made by interviewees that were extracted from interview transcripts	Interviewee (I) who made the statement	Transcript page number (separate documents - available on request)
“Microsoft recommends that you patch every Tuesday; that’s not practical in a corporate such as ours, you’d probably need a dedicated patch management team and you know here we sort of use our resources for everything possible”	I1	6
“Cause it’s happened in the past and I’ve experienced it where we patched something; it’s broken IE to hell and gone”	I1	6
“It is a combined effort with the operations teams involved so and the server team obviously, there’s the desktop support team and, uhm, the security department as well so those are the people that need to basically define that policy”	I1	9
“There are times when we do patch and then for a month or so we would actively be busy and then eight months could go by and no-one talks about patching”	I3	1
“I know for a fact with XP we never patch, we did years ago then we fell behind and then we just stopped, uh, we are hoping now to change this with Windows 7 deployment”	I3	1
“I haven’t seen the policy but it is somewhere on the intranet I believe”	I3	1
“Honestly I think everyone is already involved that that should be, uhm, but I didn’t or I don’t think it’s defined properly; I think names should be associated to certain roles and people should be held accountable for their responsibility”	I3	2
“Our biggest challenge, sorry, is that we are geographically scattered and we don’t have the infrastructure that we would wish to have”	I3	2
“We don’t have a formal mechanism for prioritising patches; we only do security and critical patches from Microsoft; if it’s a patch that fixes an active issue in our environment then we deploy as soon as possible; normally within a week”	I3	3

Quotes made by interviewees that were extracted from interview transcripts	Interviewee (I) who made the statement	Transcript page number (separate documents - available on request)
“Recently in the beginning of the year we deployed a JAVA patch tested on the virtual machine; all seemed fine, deployed to a pilot of five, no communication made it back to us, interesting thing happened here - three of these five people logged a call because on a web based application because a web based application no longer worked, desktop support did a system restore, removed the patch and the issue was resolved; all this time we were under the illusion that we have a hundred percent success rate so far, the next week we deployed to thirty people, uhm, then it became an issue because thirty users could now not use the realist system, so here although we did test, our communication strategy failed us”	I3	4
“The policy should reflect the importance of patching, people don’t understand it and therefore they simply ignore it so the policy should say why we need to patch, the policy should define roles and responsibilities and furthermore it should state the frequency of patch deployments, it should also state the scope of what we need to patch we may choose not to patch everything”	I3	4
“I think security should develop the policy but all parties involved must be consulted for input”	I3	4
“The importance of patching is not always well understood”	I3	1
“You have to align with the application vendor of the application”	I4	1
“It’s easier if you run typical Microsoft workload for instance, so if you run an exchange server or a domain controller, activating domain controller, to patch them consistently, that’s typically not an issue because its Microsoft with Patches for Microsoft, but where you get challenges is when you running third party applications”	I4	1

Quotes made by interviewees that were extracted from interview transcripts	Interviewee (I) who made the statement	Transcript page number (separate documents - available on request)
"We don't prioritise where we say every six months we patch with the latest critical and, uhm, you know security patches that's our policy so we don't prioritise in that, however if we get for instance we hit a snag or an issue, uhm, be it anti or virus related, malware related, uhm, you know because sometimes there's windows security patches also for a vulnerability that gets highlighted then typically our governance, our xxxxxxx's *[name omitted]* team, our security and governance team, they're responsible and they will bring it under our attention"	I4	4
"We've rather had issues patching servers, to be quite honest with you, than not patching servers so the counter is actually more true in my practical experience"	I4	5
"There's definite risk also in an active patching strategy because you've got Legacy applications"	I4	6
"If you haven't patched for long period like where we started a year ago, then your immediate risk is that the server has been running for seven years like this and now you patch it; there's about 85 some of our servers had about a 135 updates that needed to apply so you're touching 135 sets of DLL's and sorts of things, so chance is actually big that you might affect something on that application, but I think once you're active, once you're in your second and third cycle, it becomes a no brainer"	I4	6
"I think we should use the practical experience of operations teams"	I4	7
"the policy should also define the technique strategy and emphasise the importance thereof and then the roles and responsibilities"	I5	4

Quotes made by interviewees that were extracted from interview transcripts	Interviewee (I) who made the statement	Transcript page number (separate documents - available on request)
“The biggest challenge for me is getting the change approved in CAB, they are very reluctant to approve because generally they perceive that patches don’t add any value although they have the ability of breaking something now; with that being said it’s very difficult to justify the deployment of patches in production servers”	I5	1
“By showing the commitment of senior management people are more likely not to push back on proposed patch deployment”	I5	2
“We do have tests; yes the problem is we can test an installation on these test boxes but we don’t have users testing the apps on the boxes, so the true test is only really when we deploy to production”	I5	3
“We only do Microsoft patches and we only do security and critical patches; obviously if a patch is fixing something that is broken then you need to apply it as soon as possible”	I5	3
“We deployed a fix to the exchange server and suddenly all outlook desktop lines started prompting for new user names and passwords; turns out this patch actually broke the Kerberos protocol, I now it’s important to know we did test this patch on our server and passed our test; it installed successfully but that’s the thing about applications - they mustn’t just install they must actually work”	I5	3
“An independent department should draw up the policy and all the role players should be consulted”	I5	4
“We are not in a position where we patch very often, we haven’t been in a, there’ve been issues in the past with applications and issues with patches and the guys just stopped patching”	I6	2
“If you’re gonna install automatically, install stuff like service packs and drivers and all other kinds of funny stuff, er, that you haven’t properly tested then you’re gonna get some issues”	I6	5

Quotes made by interviewees that were extracted from interview transcripts	Interviewee (I) who made the statement	Transcript page number (separate documents - available on request)
“One of our biggest challenges is time slots; at the moment business needs always to precede patch deployments, our second challenge is bandwidth; we simply don’t have the infrastructure to push patches to every single store”	I7	1
“I know security is busy drafting a policy but the whole process is still at its early stages”.	I7	2
“I do feel like roles and responsibilities should be clearly defined”	I7	2
“Deployed a bug fix for IE8. This was still on XP and the install prompted for a reboot and all the machines gave a blue screen error; fortunately we created regular restore points so we could roll back easily”	I7	3
“our biggest challenge in the stores environment is getting the time slots to do it because there’s so much change being driven in the environment by the business”	I8	2
“The business changes always trump the patching, so if we say we’re gonna do patching on this day, it’s fine until there’s business requirements“	I8	2
“I think security ought to be involved more than they are, uhm, even if it’s just not necessarily from a doing point of view but from a motivating point of view”	I8	3
“I think security should be standing up and motivating and uh saying that no we have to patch it’s a requirement and you know we’re at as much risk if we don’t patch as if we do patch so uhm I think that they should be playing their governance role more than they are in my opinion”.	I8	3
“More parties need to understand patching, uh, they need to understand the implication of patching or the implication of not patching”	I8	3
“I haven’t actually seen the patch management policy”	I8	8

Quotes made by interviewees that were extracted from interview transcripts	**Interviewee (I) who made the statement**	**Transcript page number (separate documents - available on request)**
"If you've got a policy that says well we have to do it then you can hold the policy up and say no sorry I can't move it because the policy says I have to do it."	I8	8
"The problem we had is this is actually the first time that they patched the store environment, [company name removed] stores domain so for two years they went without patching so this is actually a very very big exercise because there would be a lot of updates and stuff a lot of updates that would have been superseded and all that stuff that you needed to bear in mind and take of when we done it"	I9	1
"We're also signing up to their patch management notification"	I9	4
"Our WSUS server which is attached to the SCCM server will then get that content for us and download it, and based on that we'll create templates".	I9	4
"We don't have that 360 degree feedback"	I9	4
"One of our issues is that we don't patch enough"	I10	2
"I know that we've had issues with software, whether it's platform software like Biztalk or sequel server, uhm, where we could have actually resolved those issues if we had been pro-active around applying patches. I also know that we've had issues where we've actually applied patches and had issues because of it"	I10	2
"Some of those mature vendors have, uhm, processes in place where they actively pick up issues and they release patches"	I10	3
"We've had a couple of incidents where we haven't tested things properly so we've actually applied a patch and there might have been an issue as the result of a patch and that's typically because we didn't test it properly"	I10	4

Quotes made by interviewees that were extracted from interview transcripts	Interviewee (I) who made the statement	Transcript page number (separate documents - available on request)
“If we don’t suffer from a problem, we typically wouldn’t apply it”	I10	4
“Microsoft patch is they come down and they then you get things like Adobe which just spring in to life and just patch every now and again”	I11	4
“We have in the past been a bit suckered where we haven’t applied patches properly and then when you do come to the point where you have to apply the patches you’re so far behind that it becomes it doesn’t become worth patching then it’s a big and expensive exercise of re-installing operating system”	I11	7
“The most frequent and visible sign is a worm-breakout when a, when you have a worm breakout on the network you can easily see which systems or work stations or which servers is the most affected and those are the ones typically that’s the worst patched”	I12	4
“I’ll only add people from an information awareness point of view”	I13	2
“I’ll only add people from an information awareness point of view but I think we first need to get the process established and off the ground before we go to that, uhm, level”	I13	2
“W-SASS only works with Microsoft so you can’t actually deploy like Adobe or semantic patches through W-SASS, so it’s only for Microsoft”	I13	3
“The problem is actually maintaining a software asset register and then following up with each and every one of those application vendors to see if they’ve got patches”	I13	3
“It’s a very well mitigated risk in our environment”	I13	4

Quotes made by interviewees that were extracted from interview transcripts	Interviewee (I) who made the statement	Transcript page number (separate documents - available on request)
“When Conficker came out but, uhm, we actually had issues where machines weren’t patched so, uhm, and then the antivirus on the machines weren’t up to date where Conficker brought down our environment and half of our store’s environment for more than two days”	I13	4