Migrating from Closed Source Software to OSS – Security Issues and Challenges

Olusegun Ajigini
UNISA, South Africa

Abstract

Open source software (OSS) allows users and developers to, amongst others, study, modify and redistribute the source code. At the other end of the spectrum, closed source software (CSS – the antonym of OSS) disallows users and developers to access and modify any source code. This paper presents an overview of the migration from closed source systems (CSS) to open source systems (OSS) with respect to Security Issues and Challenges. The paper proposes a Model to address such challenges. The various OSS initiatives undergone by different national governments are highlighted. A comparison of OSS security and CSS security aspects, and an overview of security challenges during migrations to OSS are presented. On the strength of these comparisons, and using summative content analysis, a model to address security challenges during migrations to OSS is proposed. The model is developed through enhancements of previous models aimed at addressing security threats and protecting sensitive information during system migrations.

1. Introduction

The perpetual rise of Open Source Software (OSS) has been a feature of the software industry during the past ten years [31]. Subsequently, there has been an increasing interest in the open source movement as a new paradigm for software development in recent years [46]. One such example is the South African Government who has been in the vanguard of using OSS since 2001 by adopting policy recommendations in 2003 [35].

The gradual proliferation of articles and reports in the mainstream media spearheaded evidence for an increased awareness of, and interest in open source [56]. This is also the view of Hoepman that there are many publications on OSS advantages and disadvantages [25]. It is estimated that at least 80% of all commercial software solutions would have had substantive open source components by 2012 [15].

A recent study conducted by the International Data Corporation (IDC) indicates that the OSS market will grow at an annual rate of 22.4% and would reach US$8.1 billion in 2013 [28]. However, it has been pointed out by Gwebu that low acceptance rates of OSS continue to reduce its share of the market [18]. In later work, it is emphasised that OSS benefits would not be fully realised until it is accepted and used by the mainstream software users [18].

The issue of the security of OSS was highlighted by two events, namely, a report released by Fortify Software in July 2008, claiming that necessary standards were not achieved by OSS developers, and that Linux kernel developers had covered up security vulnerabilities [34] [40]. It was recommended in this report that OSS should be viewed warily due to alleged high risks involved by government and commercial organisations. The report further recommended the conducting of risk analyses and code reviews on any OSS code running in business-critical applications.

According to Manfield-Devine, the US Department of Homeland Security as part of the US Government's Open Source Hardening Project, backed Coverity Software to investigate security issues affecting OSS products, and they came up with a report that disagreed with the Fortify findings [34] [39]. Coverity Software analyzed 55 million lines of code across 250 (amongst other Linux and Apache) projects and concluded that OSS quality and security are improving.

OSS should be evaluated from a security perspective to ascertain the level of security robustness or potential exposure to threats [3]. They also stressed that the increasing use of OSS may pose several security challenges to organisations.

In this paper we propose a model to address a number of security challenges during the migration from CSS to OSS. Our model is constructed from the Aner and Cid framework proposed and part of the rudimentary framework to protect sensitive information during system migrations [1] [3].

The layout of the paper follows: In the early sections, we consider some definitions of closed source software (CSS) and open source software (OSS). We also highlight the history of OSS, OSS initiatives, OSS projects and OSS Migrations. This is followed by a comparison of the possible benefits of OSS over CSS. In a similar vein, the second to the last section considers and compares security aspects in both CSS and OSS. The last section presents an overview of some security challenges in migrating from a closed source to an open source environment, followed by our model aimed at addressing some of these challenges. The paper concludes with references to future work in this area.

2. What is Open Source Software?

Open source is described as software of which the source code is distributed along with the executable program [20]. Such software is free to use and it includes a license allowing users and developers to modify and redistribute the software. OSS refers to any software that is distributed under OSS licensing formats [19] [59].

Open source code is accessible to the users and such code may be enhanced for added functionality [59]. Possible errors in the code can be corrected and overall improvements to the source code can be done. Open source is a term of art which is the opposite of closed source in the sense of having its source code freely available for anyone to make enhancements or correct errors [43].

A comprehensive definition of open source software is defined by the Open Source Definition (OSD) and they list eight requirements of OSS as: (a) Free Redistribution, (b) Source Code availability, (c) Derived Works, (d) Integrity of the author's source code, (e) No Discrimination against Persons or Groups, (f) No Discrimination Against Fields of Endeavor, (g) License Must Not Restrict Other Software, (h) License must be Technology-Neutral.

There are many classifications of open source and closed source in the literature [19] [20] [59]. In this paper, we define open source software as software of which the source code is distributed along with the executable program, having the right of redistribution, open standards, free to use, but could be paid for (e.g. paying for the medium on which such software is distributed) as illustrated in Table 1.

In Table 1, our definition for open source covers both Quadrants A and C. Examples of open source and closed source programs are listed in each of the four quadrants. For example, Ubuntu Linux is a free, open source program in Quadrant A, while MS Office is closed source software that is paid for and resides in Quadrant D.

Table 1. Classification of Open and Closed Source Software

	Open source software	Closed source software
Free	**Quadrant A** e.g. Ubuntu Linux, Apache	**Quadrant B** e.g. Internet Explorer, Adobe Acrobat Reader
Paid For	**Quadrant C** e.g. Red Hat, MySQL	**Quadrant D** e.g. MS Office, MS Windows operating systems

3. What is Closed Source Software (CSS)?

Closed source software (CSS) is a term invented as an antonym for OSS and is used to refer to any program whose licensing terms do not qualify as OSS. This implies that a user will have the binary version of the software that they are licensed to, without any copy of the program's source code. A user of closed source software cannot render modifications to the software. CSS is based on the assumption that software development is a highly specialised process that is managed by a team of specialised developers and best practices project management, all of which result in new releases and enhancements from time to time [46].

In this paper, we define closed source software (CSS) as software that users cannot render modifications to and can be free or paid for as illustrated in Table 1. An example of closed source software that is free as shown in Table 1 is the Internet Explorer which can (e.g.) be downloaded from the Internet.

4. History of OSS

The foundation of the OSS movement can be traced back to the US academia of the 1960s, when there was a cultural attitude of opposition to the restrictive nature of exclusive rights under intellectual property laws [31]. Richard Stallman, an ex-MIT academic launched the Free Software Foundation (FSF) in 1985, which is dedicated to the development of free software as a non-profit body [31].

Richard Stallman is considered to be the founder of the open source movement because he holds very strong philosophical beliefs that all users of computers should have the freedom to enhance any software in order to support their needs and also to share software [43]. He started to write the GNU software (an acronym for 'GNU's Not Unix). He also developed a licencing system for GNU software called 'copyleft'. The FSF was set up to further the development of the GNU software [43].

The FSF freedom philosophy is openly anti-business and this concept was promoted to a wider business community in 1997 by a group of free software community leaders [43]. This group came up with the name 'open source' and they came up with a definition to provide the requirements for open source software. This is the group that created the Open Source Institute (OSI) that manages the Open Source Definition (OSD).

According to Kemp, the FSF oversaw the GNU project, which was a mass collaboration, to create a free full operating system that will replace the UNIX system under the GPL – the GNU General Public

Licence (GPL) [31]. In 1992, the operating system kernel (known as GNU Hurd) had not been completed, though all the other necessary components had been completed. The GNU software was combined with Linux in 1992 (a new kernel) to have a complete operating system, a combination known as GNU/Linux and licensed under the GPL.

Linus Torvalds, a 21-year-old Finn and a computer scientist at Helsinki University, developed Linux, which was firstly publicly released in September 1991 [43] [31]. The name 'Linux' is derived from his first name and 'UNIX'. Linux is freely available over the Internet and suggestions for enhancing the system are requested from the public. Linux is being used and adopted commercially by many computer manufacturers and it is a competitor for the Microsoft Windows operating system – a closed system [43].

According to Kemp, the Open Source Initiative (OSI) was established by Eric Raymond and Bruce Perens in 1998 to promote OSS on pragmatic grounds [31]. Part of the function of the OSI is to review and approve licences conforming to the Open Source Definition (OSD) which was carved out from the OSI (Open Source Initiative). The OSD has ten requirements that must be adhered to before software can be allowed to be classified as open source.

5. OSS Initiatives

This section describes various OSS initiatives undergone by different national governments (South African and foreign governments). The use of OSS gained momentum in the last decade in both public and private organisations [62]. Internationally, governments see OSS as a tool that can assist them in enhancing affordable service delivery due to its low cost of implementation and maintenance [37]. However, Oram reiterates that procuring OSS has proven difficult in governments and it is difficult to get information on government usage of OSS [41].

Some studies have shown that OSS has a tendency to be cheaper on cost but more expensive on consultation and maintenance [9] [14] [21] [37] [44]. The close to zero license costs of OSS does not necessarily translate to lower costs [50].

5.1. South African Government Initiatives

The South African Government has strongly expressed the desire to use FOSS since 2001 [35]. The South African cabinet accepted two FOSS policy submissions, one by the National Advisory Council on Innovation (NACI) in 2002 and the other by the Department of Arts and Culture, Science and Technology in 2003 [63]. The Government IT Officers (GITO) Council FOSS Working Group compiled the 2003 FOSS policy for government (Cabinet Memorandum No. 29 of 2003) and this encouraged the use of FOSS in the SA Government [63].

A FOSS policy was approved by the South African Cabinet in 2007, stipulating that all future software should be based upon open standards and encouraged the migration of current government software to FOSS [16] [63]. The State Information Technology Agency (SITA) with the Council for Scientific and industrial Research (CSIR) established a project office (FOSS Programme Office) that will oversee the implementation of this policy [63]. Despite these decisions of government, FOSS adoption has not met its targets [63].

The South African government started implementing FOSS within its departments since 2006 and has a target of 60% for back-end servers running FOSS [60]. However, the results of a survey conducted from November 2007 to March 2008 suggest that FOSS is not yet widely used within the South African government [63]. They conclude that FOSS implementations in the SA government are rather few.

The South African government is still using FOSS in the development of their software systems. For example, FOSS components are used to develop the Integrated Financial Management System (IFMS). This was done to lower the cost of supporting the software (e.g. license costs) and also to improve on the quality and productivity of the IFMS software. FOSS is being used by South African government departments and many FOSS migrations have been performed by government departments.

5.2. Foreign Government Initiatives

This section outlines some of the various foreign government initiatives on the adoption and implementation of FOSS in their organisations.

i. Indian Government: The Indian Government supports the use of FOSS and has clear policies in this regard [35]. India has implemented many projects in support of FOSS adoption [51]. FOSS implementations have been carried out in many countries, e.g. China, Pakistan, and the South Americas [23] [45] [64].

ii. Malaysian Government: The Malaysian government provided comprehensive implementation guidelines for FOSS adoption and approximately 128 Malaysian state agencies migrated desktop users to FOSS by March 2008 as detailed in the Malaysian Public Sector Open Source Software Master Plan [57] [58].

iii. Brazilian Government: The Brazilian government also implemented and adopted FOSS, and has a large number

of FOSS developers and contributors [27] [36]. According to SERPRO, almost 60% of state departments in Brazil were using FOSS in 2005 [49]. A group of Brazilian proponents of social change joined the FOSS communities and accelerated FOSS adoption by many Brazilian Government Agencies during the earlier part of the Lula Administration [52]. The competence of IT professionals' impacts on the Brazilian FOSS adoption and the use of FOSS in Brazil have sky-rocketed because many Brazilian educated professionals are committed to FOSS.

iv. German Government: The German government also implemented many FOSS projects: migration from MS Exchange 5.5 to KOLAB, migration of 14000 Windows desktop and laptop computers by the Munich Municipality in 2004 to Linux and OpenOffice.org, migration of 10,000 desktop machines by the German Foreign Office to FOSS across 300 sites in 2007 [33] [38] [42]. The central administration of Germany signed an agreement with IBM to supply FOSS products based on Linux at a reduced price [37].

v. US Government: The US Government launched its recovery .gov Website known as Drupal and it was based on an Open Source Content Management System [48].

vi. British Government: The British government adopted a policy on FOSS in 2002 [37]. The objectives of this policy include the use of products based on open standards, and avoiding problems of over-dependency on a specific supplier. The policy enhances the use of FOSS in all publicly funded British organisations (Central Government Departments and their Agencies), local governments, non-departmental public institutions, the National Health Service (NHS) and the educational sector.

vii. French Government: France set up the Agency for Information and Communication Technology (AICTA) in 2001 and it facilitates the use of FOSS by public agencies [38].

viii. Spanish Government: The Spanish Ministry of Industry, Tourism and Trade gave financial support for FOSS implementation to various government institutions and autonomous administrations [4]. Some FOSS implementations include GNU/Linux, Guaclalinux, Guadainfo, Linkat, Council of Zaragoza and MAX.

Many foreign governments migrated to FOSS in order to lower the cost of supporting the software (e.g. license costs) and also to enhance the quality and productivity of their systems. Protecting sensitive information during migrations is a way of improving the quality of a software system and the research reported on in this paper may well assist foreign governments to improve on the quality and productivity of their software migrations with the understanding of the security issues and challenges during such migrations.

6. Most Popular OSS Projects

The emergence of open source software (OSS) is a recent major development in Information Technology [6]. They point out that the commonly used OSS products are MySQL database, Apache web server, the Firefox web browser, the Linux operating system, the Openoffice office suite and the DRUPAL content management system. They maintain that the business value that OSS brings to organizations include tangibles such as cost savings and reliability as well as intangibles such as innovation and flexibility. Software innovation has been democratized by OSS [2]; however, there are doubts whether this innovation can be used in business applications where the end users are not the individual developers.

Table 2 provides short descriptions of some important OSS projects cited by different authors.

Table 2. Examples of Important OSS projects

OSS Projects	Project Description	Author
Linux	Linux is the leading enterprise server operating system currently with more than seven million users.	[46] [31] [54] [6]
Perl	It was designed by Larry Wall. It is the software supporting the most 'live content' on the Internet. It is used by more than one million users as at 2005.	[46]
Python	It was designed by Guido van Rossum. It is used to integrate systems more	[46]

	effectively as a scripting or glue language that connects existing components together. It is used for rapid application development and utilised by over 325,000 users.	
GNU Project	It is a set of high quality programming tools from the Free Software Foundation's GNU project.	[46]
Apache	It was originally created by Rob McCool in 1995. Currently, it is used in over 53% (1 million) of today's web servers which is more than Microsoft's IIS at 23%.	[46] [31] [54] [6]
Mozilla	It is the open source version of its popular browser product Communicator and it is Netscape's next-generation web browser. Nescape produced the first commercially successful web browser called Navigator in 1994 as closed source software. Netscape decided to make the Navigator source code freely available under a project known as Mozilla in 1998 because the Microsoft internet explorer was made free.	[46] [31] [54] [43]
Ghost-script and Ghostview	Ghost-script is a postscript editor and printer. Ghostview enables the viewing of postscript files.	[46]
Open-Office.Org	It is the open source word processor, spreadsheet and other general-purpose office application software from Sun Microsystems.	[46] [6]
Drupal	It is a content management system	[6]

OSS is being developed and used by companies like Google, eBay and presently Facebook [34]. This implies that the distinction between OSS and closed software might not be crucial because the OSS vendors are now the commercial enterprises. The services support, and guarantees of continued development given by major OSS distributors such as Red Hat are the same as the closed source software vendors [34].

The same development tools, practices and at times, the same developers are being used by both OSS and closed source vendors [34]. Tools and processes used by OSS vendors include: public bug tracking, regression tests, security architecture review, code-scanning (simple pattern matching, or static analysis), systems tests, and penetration testing. Many closed source software vendors use these tools because they are well known and trusted.

7. OSS Migrations

According to Oram, the various successful OSS migrations include: (a) Migration from Microsoft Office to OpenOfice.org and also from Windows Operating System (OS) to Debian GNU/Linux by Munich; (b) Migration to OSS by a Brazillian stratum of educated professionals; (c) Migration to OSS by OSS advocates and civil society organisations and (d) Migration from Microsoft Office to OpenOffice.org in mid-2000 by Massachusetts, USA [41].

The factors governing the sustainability of OSS Migrations have been investigated using a qualitative and thematic analysis approach [29]. The work of the Shuttleworth Foundation has increased the knowledge and importance of OSS in South Africa in recent years [29]. Several South African municipalities have migrated to OSS with various levels of success [24].

A survey about OSS usage at universities and research centres indicates that 60% of servers; 42% of database systems; 67% of email systems and 87% of tools for managing contents of universities are

based on OSS [5]. Research on characterising OSS migration initiatives has been performed [22]. They found that software migrations from proprietary to open source depend on organisational and contextual factors such as the IT resources accessibility, organisational climate, organisational complexity, political support, why the change is needed and the project leadership style.

An overview of OSS migration and criteria for migration challenges has been presented [17]. He points out that organisations migrate to OSS from legacy systems because the legacy systems are difficult to integrate with the newer technologies. The OSS migrations can include:

- Language or code migrations;
- Operating systems migrations;
- Data migrations;
- User interface migrations;
- Architecture migrations.

8. Benefits of OSS vs. CSS – A Comparison

Table 3 is a comparison of the benefits of OSS and Closed Source Software by different authors. This Table reveals that there are more profound benefits of OSS than for closed source software.

Table 3. Comparing the Benefits of OSS and CSS

Benefit / Characteristic	Open Source Software (OSS)	Closed Source Software (CSS)	Author
Reliability	OSS has increased reliability over closed source software. The reason is that OSS is usually critically examined by many independent and enthusiastic developers during all its developmental stages.	The reliability of some closed source software is lower than that of OSS. The reason is that CSS is produced by a smaller number of developers who work against tight deadlines under much pressure.	[43] [12] [13]
Sense of Urgency	There is little sense of urgency in OSS projects; there are little or no strict deadlines, and no hierarchical team structure in OSS developments.	Due to stringent deadlines to be met, there is a sense of urgency of CSS projects. There is a hierarchical team structure in closed source projects – the corporate world.	[30]
Quality	The quality of OSS is perceived to be higher than that of CSS. This is because many developers examine the software, facilitating the detection of errors.	CSS is perceived to have a lower quality than OSS. Developers outside the closed group cannot detect errors because the source code is generally not publicly available.	[12] [32] [13]
	The quality of OSS products should be higher than for CSS if there *is* competition between them in the market	Quality of CSS could be higher than quality of OSS if there *is no* competition in the market.	[46]
	Generally there are no formal inspections in the quality of OSS programs and no broad testing. There is little evidence to support rigorous measurements in OSS.	There are formal inspections conducted in CSS projects as well as broad testing. Rigorous measurement is performed in CSS imple-mentations.	[30]

Innovation & Flexibility	OSS has more flexibility than CSS – source code is publicly available. By providing users with the freedom and flexibility, OSS enables innovation to modify the software without any restriction.	CSS has less flexibility than OSS due to its code being closed. Users are not allowed to see the source code and this restricts innovation. But it facilitates the security and reliability of the software. They have targeted innovation that is business focused rather than technology focused.	[12] [13] [8]
Software Requirements	Requirements are mostly absent in OSS projects. There is little systematic effort in addressing Capability Maturity Models (CMMs). There is also little evidence of using the Unified Modeling Language (UML) or any form of systematic modeling in OSS.	Requirements are used in CSS projects. The Capability Maturity Model (CMM) is well addressed in CSS projects. Closed source projects make use of UML or other modeling techniques.	[30]
Vendor-Lock Ins	There is no Vendor-Lock In associated with OSS. The user is independent of the vendor.	CSS is dependent on the Vendor. Therefore, there is vendor-lock in.	[12] [13]
Cost	OSS tends to be free; and have low acquisition cost, except for having to pay for the media on which the software may be distributed (e.g. on a CD). The total cost of ownership may roughly be the same as for some closed source programs.	Most CSS are not free and have a higher acquisition cost than OSS. However, in some situations closed source Total Cost of Ownership (TCO) is lower than that of open source. TCO for closed source and open source software could roughly be the same.	[12] [13] [59] [46] [8]
Adherence to Standards	The use of standards is limited to data formats like the Hypertext markup language (HTML), or the Extensible markup language (XML).	Closed source projects normally adhere to most IT standards during implementation.	[30]
Usability / Ease of code errors identification and problem solving	Most OSS products offer code error reporting tools. These tools assist in the faster detection of errors and the rapid finding of solutions.	Generally, it requires a much longer period to resolve errors in CSS, due to non-availability of code error reporting tools.	[59]

	OSS usually lacks usability because it is developer-centric. Ability to correct errors is limited to users with technical expertise.	Closed source programs do not lack usability. They employ expert usability testing techniques and usability is ranked quite higher than in OSS.	[8] [32]
Operating Systems	OSS products are supported with operating systems that surpass the operating systems that support CSS due to the availability of source code which can be altered. Users can adapt the OSS to their operating systems. The cost of such a diversity of operating systems tends to be higher in closed source systems due to their high development costs.	It is more expensive to change the operating system source code of a CSS. Development costs are generally high. Users usually have to wait for a next release of the software.	[59]
Documen-tation	Most OSS projects are weak on document-tation. OSS are not legally bound to produce document-tation such as manuals or guides.	Most CSS projects produce manual and quality documentation. Closed source programs are legally required to supply documentation such as user manuals, guides etc.	[59] [13]
Personali-sation	This is the degree to which developers are able to write applications in the way they want the application to look and are used. OSS developers use personali-sation a lot in their work in order to change the look and feel of a product, so that it can integrate seamlessly with their working environment. This enhances their efficiency and mood.	CSS developers are generally not allowed to attach personalisation to their work. Company standards and policies have to be adhered to and CSS is designed to accommodate the generic software market.	[59]
Service and Product Support	OSS products come with many learning materials obtainable from the developer's site or other locations supporting the OSS product. Large community of users and developers support OSS products by designing tutorials and short articles on	Closed source systems are supported by a support team and they usually make use of printed material or books which come at a cost.	[59]

	how the product should be used. User groups are available and support is delivered via forums and blogs. Issues may, or may not be resolved soon.	Closed source programs have a high response service. Ongoing support is provided to the customer. Support to the users of CSS is arguably the greatest advantage of using CSS.	[8]
Plug-in functionality	Is readily available for OSS products. OSS developers and users can extend the functionality of their product by using Plug-ins to write their own modules which can be integrated with the OSS product.	It is more difficult to write Plug-ins for Closed Source systems than OSS because documentation is not as rich as the OSS. The source code is also not readily available.	[59]
Highly specialised Applications	OSS programs are less likely to be used to develop highly specialised applications. There is little evidence that formal specifications are used in OSS projects and this limits the use of OSS in safety-critical software.	CSS can be used effectively to develop highly specialised applications. Formal specifications are used in closed source projects and this enhances their use in safety-critical software.	[46] [30]
Best-practices Project Management	PM practices are usually lacking in most OSS projects and this could undermine the product's quality. Release management guidelines are informal in OSS and there are often version proliferation and implement-ation issues.	Most closed source projects use best-practices project management techniques, all of which enhance a product's quality. Most closed source projects follow release management guidelines.	[46] [30]

Discussion of Table 3

The reliability of some CSS may be lower than that of OSS owing to fewer programmers that develop closed source software, working against tight deadlines and under a fair amount of pressure [12] [13] [43]. Closed source software is perceived to have a lower quality and lower flexibility than OSS due to the non-availability of the source code [12] [13] [32]. However there are arguments that CSS is of a higher quality than OSS, provided that there is no competition in the market [30] [46].

Most CSS implementations make use of a modeling language like Unified Modeling Language (UML), as well as incorporating the Capability Maturity Model (CMM). In contrast, OSS implementations usually do not make use of any modeling techniques like UML; neither do they use the CMM [30].

The Total Cost of Ownership (TCO) of both OSS and closed source software are roughly the same [8]. Closed source programs do not lack usability, documentation or service/product support, whereas OSS programs usually lack usability and documentation [8] [30]. There is no vendor lock-in associated with OSS but closed source software is characterized by vendor lock-ins [12] [13].

According to Raghunathan, the comparisons of open source and closed source are not conclusive, or in a finer analysis are slightly in favour of open source [46]. This is also the view of Khanjani, namely, that OSS yield more benefits than CSS [32].

More enthusiastic developers are involved in developing, testing and evaluating the code of OSS programs.

9. Comparing OSS and CSS Security

The importance of analyzing a whole OSS system when performing an extensive security investigation has been emphasised [20]. Such analyses include the application software, its source code, and the tools used for developing the object code. Examples are compilers, operating systems, hardware and the whole development environment.

Different authors have different perceptions when they compared OSS security with that of CSS as shown in Table 4. The table reveals that the security of OSS is roughly of the same quality as that of a CSS system.

Table 4. Comparing OSS and CSS Security

Characteristic	OSS security	CSS security	Author
Publishing of Designs and Protocols	OSS designs and protocols are published and these contribute to the security of the systems. This may reveal logical errors in the security of the system.	Closed source designs and protocols are not published.	[25]
Finding and correcting security vulnerability	It is easier to find and correct code errors in OSS than in CSS owing to the openness factor.	Open and closed approaches to security are rather similar. Correcting errors in CSS is dependent on the programming team that developed the program – the source code is not publicly available.	[10]
Checking and Testing of code	OSS users have the freedom to validate and test the code of the OSS product that they want to use so as to ascertain its quality and security.	Because users do not have the choice to validate and test the code in closed systems, the author stresses that OSS initial coding tends to be of a higher quality than CSS.	[34]
Controlled Environment Development	OSS is often viewed as having security issues because OSS is not necessarily developed in a controlled environment.	CSS is perceived as being more secure because it is developed in a controlled environment by a concentrated team with a common direction. The source code may be viewed and edited only by this team. The software is comprehensively audited, eliminating the risk of back door Trojans and reducing the risk of code errors or other software issues.	[8]
Closeness or openness of software code – security through obscurity	It is maintained that OSS improves software transparency, security and trustworthiness because users and developers can validate an OSS program's functionality and security, due to the availability	The authors stress that the security of software is dependent on the user and not necessarily its closedness or openness. CSS can also be as secure as OSS.	[20]

	of its source code. They highlight that it is easier to correct bugs in OSS systems thereby enhancing the quality of code. This could also lead to the use of better project management and quality control. Open source users can independently evaluate the security for themselves. The real exposure of the system can be assessed and the gap between perceived and actual exposure is diminished.	CSS does not allow users of such software to evaluate its security for themselves. This does not allow users to easily discover weaknesses and 'patching' is not possible by users.	[25]
Analysis of published vulnerabilities	There are no significant differences in terms of vulnerability severity found between open source and closed source. More and faster patches can be found in open source systems. Patches for open source systems are released faster than for closed source systems.	The vulnerability severity found between open source and closed source are perceived to be the same. Patches for vulnerabilities of closed systems are released weeks or months after the discovery of the vulnerabilities and this increases the risk of using the system.	[47] [25]
	Patch management is harder to coordinate in open source systems because OSS comes in many different versions. Patches will not be available for some distributions and they may be vulnerable to attacks while others are being patched. OSS products are more secure than CSS products. However, their general pattern of vulnerability detection is similar.	The authors claim that it is easier to manage patches in a closed source system than in an open source system. CSS products are less secure than OSS products.	[7] [61]

Discussion of Table 4

Closed source designs and protocols are not published, whereas the OSS designs and protocols are published enhancing the security of OSS programs since logical errors may be revealed [25]. This is also the view of Dwan that due to the openness of OSS code, it is easier to find and correct errors in OSS than in CSS [10]. This is also pointed out by Hoepman that more and faster patches are found in OSS whereas patches are not released as fast in CSS, thereby increasing the risk of using the system securely [25].

OSS users have the freedom to validate and test the code in order to ascertain its quality and security, therefore OSS initial coding tends to have higher quality and security than CSS [34]. However, Daniel

argues that CSS is perceived to be more secure than OSS because it is developed in a controlled environment by a dedicated team of developers with a common direction [8].

The view of Hansen is that CSS can be as secure as OSS because the security of software is dependent on the user and not on its openness or closedness [20]. The severity of vulnerabilities found between OSS and CSS are similar as pointed [47]. While our view is that OSS is more secure than CSS, there are, however, security challenges that have to be overcome when migrating from a closed system to an open system [17].

10. Security Challenges during Migration to OSS

A list of items that can be migrated is presented by Geetha and these are: (a) Language or code migrations, (b) Operating system migrations, (c) Data migrations, (d) User Interface migrations and (e) Architecture migrations [17]. He points out that the challenges to migration from Legacy systems to OSS include: (i) Qualification and selection of OSS, (ii) Human factors such as: Fear of the new software; Knowledge is power; Cost of training personnel for the new tools; reduced productivity of the personnel and (iii) Technical challenges. The technical challenges include: Usability; Software Development Service and support; Security; Data migration; and OSS Code Maintenance and Management [11].

According to Geetha and ElHag, the security challenges during migration to OSS are: (a) Detecting security risks, bugs, and errors, (b) Eliminating the bugs and errors and (c) Obtaining metrics for measuring software security for real-time and mission critical software [11] [17].

11. A Model for Addressing the Security Challenges during Migration to OSS

Summative content analysis was used as the research method to explore the model for addressing the security challenges during migration to OSS. During summative content analysis, the keywords (derived from review of literature) are identified before and during data analysis [26]. Keywords are extracted from the literature and mostly from the two articles written by Anner and Ajigini [1] [3]. An open source assessment framework and a threat modelling methodology, pioneered by Microsoft since 1999 have been highlighted, this is then proposed by Anner to overcome the security challenges of OSS [3] [53]. The aim is to reduce the risks to confidentiality, integrity and availability and to identify and reduce threats, vulnerabilities and risks to an acceptable level. They mention that alternative methods to reduce risks include: (a) Code auditing (b) Penetration testing, and (c) Using Statistical analysis tools.

As per Anner, the threat modelling process consists of four stages, viz: (i) Application Analysis/Diagramming (ii) Threat Enumeration, (iii) Threat Rating, and (iv) Mitigation Options [3]. They point out that the threat modeling approach with slight modifications can assist with the identification of security vulnerabilities, as well as investigating coding issues and implementation mistakes.

A Rudimentary Management Framework to protect sensitive information during the migration to an open source system is suggested [1]. The model we propose in this section for addressing the security challenges discussed in this paper in migrating to OSS, is based in part on the threat-modeling framework in Anner and the sensitive information migration framework [1] [3].

Our model is illustrated in Fig. 1 and is discussed below:

During the Application Analysis/Diagramming phase (A), the applications are analyzed from a flow of data perspective. All the aspects that make up the applications are catalogued and the relationships between the assets in terms of data exchange are identified through a UML Class-oriented structure.

The Threat Enumeration phase (B) consists of analyzing each element in the Class-oriented UML against a list of potential threats depending on the element type using the STRIDE Taxonomy [28]. STRIDE is used as a classification schema to characterize known threats in accordance to the attacker motivation.

The risk levels for each of the enumerated threats are determined and ratings of all threats are established during the Threat Rating phase (C).

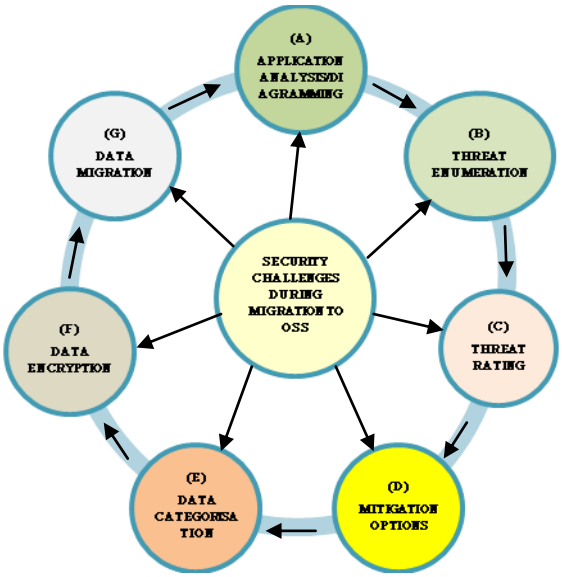

Figure 1. Modelling security challenges during OSS migration

During the Mitigation Options phase (D), all functionality and patching are removed and other security controls are added and redesigned.

The business rules and the data classification system are used to classify migrated data during the Data Categorisation phase (E).

Data protection tools and Privacy enhanced technologies are used to encrypt the data during the Data Encryption phase (F).

The encrypted data is now migrated during the Data Migration phase (G).

Implementing the Proposed Model

The following processes are proposed to implement the model in Figure 1:

Phase A: Application Analysis/Diagramming Phase –

a) Identify security objectives – user identity protection, privacy and regulation, availability guarantees of applications.
b) Catalogue all the applications.
c) Analyse all the application designs and architectures to identify the components using Data Flows.
d) Identify UML component diagrams.
e) Identify the relationships between the assets using data exchange by using Class-oriented UML structures.

Phase B: Threat Enumeration Phase –

a) Analyse each element in the Class-oriented UML diagram against potential threats by using the STRIDE Taxonomy.
b) Analyse data movement across trust boundaries (e.g. from Internet to Web tier).
c) Identify the features and modules with a security impact that need to be evaluated.
d) Investigate how data enters modules, how modules validate and process the data, where the data flows to, how the data is stored and what fundamental decisions and assumptions are made by the modules.

Phase C: Threat Rating Phase –

a) Identify threats using Bugtraq tools and techniques. Bugtraq is a mailing list containing information on how to exploit and use intrusion detection systems vulnerabilities in defending networks.
b) Determine the risk levels of each threat.
c) Establish the ratings of all the threats.
d) Use either a threat graph or a structured list to write out the threats.

Phase D: Migrations Options Phase –

a) Remove the functionality and patching.
b) Add other security controls.
c) Redesign other security controls.

Phase E: Data Categorisation Phase –

a) Develop business rules.
b) Develop a Data classification system.
c) Classify data based on business rules and the above data classification system.

Phase F: Data Encryption Phase –

a) Deploy Data Protection Tools.
b) Deploy Privacy Enhancement Technologies.
c) Use secure Tools to encrypt the data.

Phase G: Data Migration Phase –

a) Ensure that data to be migrated are encrypted by using verification techniques.
b) Migrate the encrypted data.

12. Conclusions

In this paper we investigated the notions of closed source software (CSS) and open source software (OSS); the security issues and challenges of migrating from CSS to OSS were investigated, we discussed the respective advantages of each and considered comprehensively the security aspects underlying each approach to software development.

A comparison of the benefits of OSS and closed source software by different authors was explored. The comparisons of the benefits of open source and closed source are slightly in favour of open source. Additionally, a comparison of OSS and CSS security was undertaken and our view is that OSS is more secure than CSS.

Using summative content analysis, the challenges in migrating from a closed system to an open system were identified, and these, together with two frameworks – one for threat modelling and another for protecting sensitive information during system migration were used to propose a model for addressing the various security aspects in migrating from an open system to a closed one [1] [3]. Our model is based on a seven-phase process as presented in Figure 1. It is anticipated that this model may be useful as a basis for mitigating the security challenges in moving from a closed (CSS) to an open (OSS) system.

13. Future Work

Future work in this area may be pursued along a number of lines: The framework proposed for protecting sensitive information during system migration has to be further integrated with the security-protection model proposed in this paper [1]. In particular the classification of sensitive information in phase 5 – Data Categorisation has to be further developed. Having implemented our

model, we have to validate it in industry at companies that have migrated to OSS, as well as those who are yet to undertake such migration.

14. References

[1] Ajigini, O. A., Van der Poll, J. A. and Kroeze, J. H. (2012) 'Towards a management framework to protect sensitive information during migrations', in Proceedings of the 2nd International Conference on Design and Modeling in Science, Education and Technology (DeMSet), pp. 6-13.

[2] Allen, J. P. (2012) 'Democratizing business software: Small business ecosystems for open source applications', Communications of the Association for Information Systems, 130 (28), pp. 483-496.

[3] Anner, Y. and Cid, C. (2010) Open-Source security assessment, Royal Holloway Series.

[4] CENATIC, (2008) 'Open source software for the development of the Spanish public administration: An overview'; www.cenatic.es (7 April, 2011).

[5] CENATIC, (2009). 'Study on the situation of open source software in universities and R&D centers', Report, Almendralejo; www.cenatic.es (7 April, 2011).

[6] Chengalur-Smith, I., Nevo, S. and Demertzoglou, P. (2010) 'An empirical analysis of the business value of open source infrastructure technologies', Journal of the Association for Information Systems, vol. 11, Special issue, pp. 708 - 729.

[7] Clark, R., Dorwin, D. and Nash, R. (2009) 'Is open source software more secure?' Homeland Security / Cyber Security; http://www.cs.washington.edu/education/courses/csep590/05au/whitepaper_turnin/oss%2810%29.pdf. (2 February, 2003).

[8] Daniel, J. (2009) 'Open source vs. closed source software: The great debate'; http://www.articlesbase.com/internet-articles/open-source-vs-closed-source-software-the-great-debate-1040071.html (1 February, 2013).

[9] Drozdik, S. and Kovacs, G. L. (2005) 'Risk Assessment of an open source migration project', In Proceedings of the First International Conference on Open Source Systems, Geneva, M. Scotto & G. Succi (eds.) pp. 246 – 249.

[10] Dwan, B. (2004) 'Open source vs. closed', Network Security, vol. 5, pp. 11-13.

[11] ElHag, H. M. A. and Abushama, H. M. (2009) 'Migration to OSS: readiness and challenges'.

[12] Fitzgerald, B. (2006) 'The transformation of open source software', MIS Quarterly, 30(3), pp. 587 - 598.

[13] Gallegoa, M. D., Lunab, P. and Bueno, S. (2008) 'User acceptance model of open source software', Computers in Human Behaviour, 24(5), pp. 2199 - 2216.

[14] Gallopino, R. (2009) 'Open Source TCO: Total Cost of Ownership and the Fermat's Theorem'; http://robertogaloppininet/2009/01/08/open-source-TCO-total-cost-of-ownership-and-the-Fermats-theorem/(ed.) (21 March, 2012)

[15] Gartner, (2008) Gartner highlights: Key predictions for IT organisations and users in 2008 and beyond.

[16] GCIS, (2007) 'Cabinet Statement.' Feb 22; www.gcis.gov.za/media/cabinet/2007/070222.htm (2 February, 2012.

[17] Geetha, S. (2012) 'Possible challenges of developing migration projects', International Journal of Computers & Technology, 3(3), pp. 463 - 465.

[18] Gwebu, K. L. and Wang, J. (2010) 'An exploratory study of free open source software users' perceptions', The Journal of Systems and Software, 83(11), pp. 2287 - 2296.

[19] Gwebu, K. L. and Wang, J. Wang (2011) 'Adoption of open source software: The role of social identification', Decision Support Systems, vol. 51, pp. 220 - 229.

[20] Hansen, M., Kohntopp, K. and Pfitzmann, A. (2002) 'The open source approach – opportunities and limitations with respect to security and privacy', Computers & Security, 21(5), pp. 461 - 471.

[21] Hauge, O., Ayala, C. and Conradi, R. (2010) 'Adoption of open source software in software-intensive organisations – A systematic literature review', Journal of Information and Software Technology, vol. 52, 1133 – 1154.

[22] Heredero, P., De, C., Berzosa, D. L. and Santos, R. S. (2010) 'The implementation of free software in firms, an empirical analysis', The International Journal of Digital Accounting research, 10(6).

[23] Hedgebeth, D. (2007) 'Gaining competitive advantage in a knowledge-based economy through the utilization of open source software', VINE: The Journal of Information and Knowledge Management Systems, 37(3), pp. 284 – 294.

[24] Hislop, R. (2004) 'Mossel Bay adopts Linux on the desktop', Electronic Government Africa, 1(1), pp. 14.

[25] Hoepman, J. and Jacobs, B. (2007) 'Increased security through open source', Communications of the ACM, 50(1), pp. 79 - 83.

[26] Hsieh, H. and Shannon, S. E. (2005) 'Three approaches to qualitative content analysis', Qualitative Health Research; http://qhr.sagepub.com/content/15/9/1277 (19 September, 2014).

[27] Lewis, J. A. (2007) 'Government open source policies'; http://www.csis.org/media/csis/pubs/070820_open_source_policies.pdf, (29 January 2012).

[28] Little, G. and Stergiades, E. (2009) Worldwide Open Source Services, 2009-2013 Forecast.

[29] James, S. and Van Belle, J. (2008) 'Ensuring the long-term success of OSS migration: a South African exploratory study', 6th Conference on Information Science Technology and Management, New Delhi, India.

[30] Kamthan, P. (2007) 'A perspective on software engineering education with open source', IGI Global.

[31] Kemp, R. (2009) 'Current developments in open source software', Computer Law & Security Review, vol. 25, pp. 569 - 582.

[32] Khanjani, A. and Sulaiman, R. (2011) 'The aspects of choosing open source versus closed source', IEEE Symposium on Computers & Informatics, pp. 646-649.

[33] Kovacs, G. L., Drozdik, S., Zuliani, P. and Succi, G. (2004) 'Open source software for the public administration', In Proceedings of the 6th International Workshop on Computer Science and Information Technologies, Budapest, Hungary.

[34] Manfield-Devine, S. (2008) 'Open source: does transparency lead to security?', Computer Fraud & Security, pp. 11 - 13.

[35] Miscione, G. and Johnston, K. (2010) 'Free and open source software in developing contexts, from open in principle to open in the consequences', Journal of Information & Ethics in Society, 8(1), pp. 42 - 56.

[36] Mtsweni, J. and Biermann, E. (2010) 'A roadmap to proliferate open source software usage within SA Government servers', Third International Conference on Broadband Communications, Information Technology & Biomedical Applications, IEEE Computer Society, pp. 430 - 436.

[37] Mutula, S. and Kalaote, T. (2010) 'Open source software deployment in the public sector: a review of Botswana and South Africa', Emerald, 28(1), pp. 63 – 80.

[38] Nagler, M. (2005) 'Open Source adoption of the German Federal Office for information security'; http://ec.europa.eu/idabc/servelets/Doc?id=21394 (25 January 2012).

[39] Open Source Report, Coverity Report, (2008); http://scan.coverity.com/report/ (1 February, 2013).

[40] Open Source Security Study, Fortify Report, (2008); www.fortify.com/ossreport.html (1 February, 2013).

[41] Oram, A. (2011) 'Promoting open source software in Government: The challenges of motivation and follow-through', Journal of Information Technology & Politics, 8(3), pp. 240 – 252.

[42] Otter, A. (2007) 'SA government gets serious about ODF, IOIL Technology'; http://www.ioltechnology.co.za/article_page.php?iArticleId=4126700&iSectionId=2888, (21 February 2012).

[43] Pearson, H. E. (2000) 'Open source licenses, open source – The death of closed source Systems?' Computer Law & Security Report, 16(3), pp. 151 - 156.

[44] Poulter, A. (2010) 'Open source in libraries: an introduction and overview', Emerald, 59(9), pp. 655 – 661.

[45] Rafiq, M. and Ameen, K. (2009) 'Issues and lessons learned in open source software adoption in Pakistani libraries', The Electronic Library, vol. 2794, pp. 601 – 610.

[46] Raghunathan, S., Prasad, A., Mishra, B. K. and Chang, H. (2005) 'Open source versus closed source: Software quality in monopoly and competitive markets', IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans, 35(6), pp. 903 - 918.

[47] Schryen, G. (2009) 'Security of open source and closed source software: An empirical comparison of published vulnerabilities', AMCIS Proceedings, Paper 387.

[48] Scola, N. (2009) 'Why the White House's embrace of Drupal matters', Personal Democracy Forum techPresident.

[49] SERPRO, (2005) 'Fast move to free software in Brazil'; http://ec.europa.eu/idabc/en/documents/5131/528, (20 January, 2012).

[50] Shaikh, M. and Cornford, T. (2012) 'Strategic drivers of open source software adoption in the public sector: Challenges and opportunities', European Conference on Information Systems AIS, Paper 237.

[51] Sharma, A. and Adkins, R. (2006) 'OSS in India', In DiBona, C., Cooper, D. and Stone, M. (eds.), Open Sources 2.0, O'Reilly Media, Sebastopol, CA, pp. 189 - 196.

[52] Shaw, A. (2011) 'Insurgent expertise: The politics of free/live and open source software in Brazil', Journal of Information Technology & Politics, vol. 8, pp. 253 – 272.

[53] Shostack, A. (2008) 'Experiences threat modelling at Microsoft', In Modelling Security Workshop, Dept. of Computing, Lancaster University, UK; http://blogs.msdn.com/sdl/attachment/8991806.ashx (2 February, 2013).

[54] Stol, K., Babar, M. A., Russo, B. and Fitzgerald, B. (2009) 'The use of empirical methods in open source software research: Facts, trends, and future Directions', FLOSS, Vancouver, Canada, pp. 19 - 24.

[55] Swiderski, F. and Snyder, W. (2004) Threat Modelling, Microsoft Press.

[56] The Guardian, 2004 'Open invitation taken up at last'; http://society.guardian.co.uk/epublic/story/0,,1362744,00.html. (2 February, 2013).

[57] Thomas, J. (2007) 'Malaysian public sector OSS program Phase II: Accelerated adoption';

http://www.oscc.org.my/documentation/phase2_launching/OSS-Phase-2Strategy-Plan-Launch.pdf (5 June, 2013).

[58] TMPSOSSSMP, (2008) 'The Malaysian public sector open source software master plan: Phase II – Accelerated adoption'; http://www.mampu.gov.my/seminar%20ict/kk2-OSS.pdf (19 February, 2012).

[59] Vintila, B. (2010) 'Citizen oriented open source security', Open Source Science Journal, 2(3), pp. 57 - 64.

[60] Vital W. (2006) 'The South African adoption of open source', White paper created by Vital Wave Consulting; www.vitalwaveconsulting.com/insights/South-African-Adoption-of-Open-Source.pdf, (20 February, 2012).

[61] Walia, N., Rajagopalan, B. and Jain, H. (2006) 'Comparative investigation of vulnerabilities in open source and proprietary software: An exploratory study', Americas Conference on Information Systems (AMCIS), vol. 108, pp. 848 - 857. Weber, T., 2004. *The success of open source.* Harvard University Press, New York, NY.

[62] Weber, T. (2004) The success of open source. Harvard University Press, New York, NY.

[63] Weilbach, L and Byrne, E. (2010) 'A human environmentalist approach to diffusion in ICT policies – A case study of the FOSS policy of the South African Government', Journal of Information Communication & Ethics in Society, 8(1) pp. 108 – 123.

[64] Yeo, B., Liu, L. and Saxena, S. (2006) 'When China dances with OSS', In DiBona, C., Cooper, D. and Stone, M. (eds.), Open Sources 2.0, O'Reilly Media, Sebastopol, CA, pp. 197 – 210.

