2004

Myth or fact: is open source software more secure than closed source software?

Daniel Saffioti
University of Wollongong, dfs@uow.edu.au

Gene Awyzio
University of Wollongong, gene@uow.edu.au

Robert B. K. Brown
University of Wollongong, bobbrown@uow.edu.au

Recommended Citation
Saffioti, Daniel; Awyzio, Gene; and Brown, Robert B. K., "Myth or fact: is open source software more secure than closed source software?" (2004). *Faculty of Engineering and Information Sciences - Papers: Part A*. 929.
https://ro.uow.edu.au/eispapers/929

Myth or fact: is open source software more secure than closed source software?

Abstract

This paper examines one aspect of quality that organizations look for when selecting software, namely security. Studies over time have indicated that security; scalability, interoperability and flexibility are important however the major issue has always been support. This has led to the sometimes inflexible concept of Standard Operating Environments (SOEs) within organisations. Whilst SOEs provide many benefits to an organisation they can leave them vulnerable to several large security risks. CERT statistics show that security incidents have increased six fold since 2000 [1]. This paper will examine trends in both open and closed software development for a number of platforms that may be reducing the overall security of software. Whilst SOEs provide a larger target for security vulnerabilities and the heterogeneity of Open Source products may provide a less tempting target recent trends indicate that open source software may be becoming as vulnerable as the better known closed software products.

Keywords

fact, source, myth, open, software, closed, more, secure, than

Disciplines

Engineering | Science and Technology Studies

Publication Details

Saffioti, D., Awyzio, G. & Brown, R. B. K.. (2004). Myth or fact: is open source software more secure than closed source software?. In D. Purdue (Eds.), AUUG 2004 Conference Proceedings - Who Are You? Identification and Autentication Issues in Computing (pp. 109-113).

Myth or Fact: Is Open Source Software More Secure than Closed Source Software?

Daniel Saffioti, Gene Awyzio, Robert BK Brown

School of IT and Computer Science, University of Wollongong

<{dfs, gene, bobbrown}@uow.edu.au>

ABSTRACT

This paper examines one aspect of quality that organizations look for when selecting software, namely security. Studies over time have indicated that security; scalability, interoperability and flexibility are important however the major issue has always been support. This has led to the sometimes inflexible concept of Standard Operating Environments (SOEs) within organisations. Whilst SOEs provide many benefits to an organisation they can leave them vulnerable to several large security risks. CERT statistics show that security incidents have increased six fold since 2000 [1].

This paper will examine trends in both open and closed software development for a number of platforms that may be reducing the overall security of software. Whilst SOEs provide a larger target for security vulnerabilities and the heterogeneity of Open Source products may provide a less tempting target recent trends indicate that open source software may be becoming as vulnerable as the better known closed software products.

1. Introduction

Traditionally software quality metrics have focused on proving that software products produced meet all stated requirements. In today's highly connected software environment the impact of security breaches is exacerbated. The inherent heterogeneous nature of open source software (OSS) helps to ensure systemic survivability as the number reported software security incidents increases. This paper discusses software quality metrics and the factors inherent in the open source initiative that help to ensure an overall quality software artefact.

2. Software Quality

The International Standards Association (ISO) defines quality as "the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs" [2]. Traditionally software quality metrics have focused on assuring programme integrity [3]. ISO definitions of quality state that a software artefact can be considered to be a quality artefact if it conforms to all requirements [2, 3]. It has also long been accepted [4, 5] that many software quality metrics can be addressed by using good programming paradigms [4]. Hartner et al [6] suggest that use of mature process models is a key factor in improving software quality again emphasising the relationship between requirements and quality.

Recognised software quality metrics include non-functional requirements including efficiency, flexibility, maintainability, performance, portability, reliability, reusability, testability and usability [3, 4]. Functional metrics for software quality may include accuracy, completeness, and self-containedness [4].

In order to characterise software quality metrics can be further divided into static and dynamic metrics [7]. Characteristics of the code structure, such as total lines of code, can be measured by static metrics. Dynamic metrics provide a means of measuring testing thoroughness based on structural and data-flow coverage [7].

In today's connected business environment the flow of information, ideas and electronic products has become critical business processes [8]. In such an environment the quality of software security and mitigation of total systemic failure needs to become one of the prime indicators of quality as loss of data through corruption or theft could lead to loss of money or business failure [9].

3. Open Source Software

The Open Source Initiative provides the following principles for open source software (OSS) licensing [10]. Open source software must provide for; free redistribution allowing for component aggregation from several sources, availability of source code and inclusion of source code in program. Open source software authors must explicitly give permission to modify and redistribute code under the same licensing rights; otherwise modified code must be redistributed with either a different name or version number. Other redistribution options include the use of patch files that can be distributed with the original source code. Licenses for Open Source software must not restrict who can use the software, where the software can be used, and other software that are co-distributed or specify all distribution components.

Due to the open nature of development a major criticism of the open source movement is that it lacks the support infrastructure of proprietary software products. However, it can be argued that open source support arrangements are more dynamic and customer focussed then some of the closed offerings [11-13]. The community based development structure frequently includes bulletin boards, mailing lists, archives etc [12] that are available to users.

Inherent in the Open Source Imitative manifesto outlined above is the heterogeneous nature of OSS. The open source model has been categorised by the formation of distributions e.g. Redhat, Mandrake etc that offer various implementations of Linux which aims to address certain consumer groups. Karels [13] suggests a major criticism of this model is that support organizations are frequently external to the developers. However, each distribution is aimed at providing quality user support at nominal cost and often the development community can provide a moderate to high level of support and will create patches and fixes for identified problems[12].

The community is important in both the development and support processes of Open Source Software [14]. This is particularly apparent in the development of Unix and other Open Source Projects such as FreeBSD, Linux etc. The community developed by Open Source software projects enables end users and vendors to get outstanding support immediately through a diverse, decentralised and democratic structure [13] suggesting that all members in the community gain access to those who create the technology thus allowing ideas to flow through the community to those who can make a difference effectively. Karels [13] suggests this process gives users the ability to participate in Darwinism (evolution). Ultimately the process of development in the open source community is open and transparent - thus making it a highly malleable product which is highly supported by other members.

A number of open source projects such as MySQL, Apache and Jakarta Tomcat now form a substantial part of enterprise computing [12, 15]. MySQL is an enterprise class database utility which has a wealth of information/ support resources accessible via public forums, documents or through advocates of the community. In addition to this consumers can opt to purchase enterprise licensing for MySQL enabling them to get the benefits of support from a vendor without sacrificing any of the open source benefits [16]. The open source model clearly demonstrates a high level of user support and commitment.

In addition to this the underlying principals followed for development by the community create dynamic/responsive solutions- at times more so then the larger vendors [12, 17]. Levanes [17] states; "The open source development methodology offers a number of advantages over the propriety model … One such advantage is that collaboration of productivity between peers is enabled, which customarily returns fast development, enhancements and bug fixes." This suggests that the transparency in the development process enables hardware manufactures, developers and end users to interact with one another thus creating a shared benefit and experience.

A particularly successful example of this is Andrew Tridgell's [18] development of Samba, which mimics the SMB protocol used to share files/resources on Microsoft Windows computers. Samba has become a de-facto industry standard in the community developed by corporations such as SGI, Apple and a number of smaller organizations and end users [18]. Such involvement by members of the community is difficult to find anywhere else in the computing industry and further enhances the idea that support is not an issue.

4. Survivability and Security

A major feature of many open source efforts is the long term survivability of software and lower total cost of ownership. The recent spate of viruses such as Blaster [19] and Sobig [20] has caused government and major organizations to rethink standard operating environments (SOE's) [21-23]. Until recently commercial solutions have formed the large bulk of standard operating environment due to the perception that higher levels of support and maintenance reduce costs of deploying the underlying technology. These recent attacks have caused the notion of 'monoculture' to becoming important causing The American Computer and Communications Industry Association recently made recommendations suggesting that the US government diversify its operating systems to reduce the effect of computer attacks [21].

It is often believed that hackers are motivated these days due to some vendetta against Microsoft. Recent studies modelling Hacker attitudes however show that hacking is largely motivated by intellectual stimulation (44.9%), improvement of Skills (41.3%) and beliefs of software distribution i.e. Open Source 33% [26]. This therefore suggests that any operating system is susceptible to attack. Recent statistics suggest the number of vulnerabilities detected in mainstream operating systems such as Redhat Linux and Microsoft Windows is much larger then other Unix variants (closed/open) [24]. This therefore suggests that heterogeneity in the industry is a positive attribute mitigating risks of such vulnerabilities by spreading it across a number of software distributions.

Security Focus Statistics [24] suggest that in the period of April 2001 - March 2002, Microsoft was the top vendor with the largest number of vulnerabilities (11.7%) with the rest spread across a number of commercial and open source offerings. It is also interesting to note that in the period from 2000 to 2001 Windows NT/ 2000 had 97 and 42 vulnerabilities whilst Windows 95/98 had 40 and 14 restively. Linux had 95 and 54 vulnerabilities reported for the same period and was the top of all the Unixes with other freely available Unixes in the 30's or teens for these respective periods [24]. These vulnerabilities suggest that current practices for minimising total cost of ownership are failing by placing more information technology infrastructure and assets in harms way. Open source solutions improve on this monoculture by providing a world of heterogeneous solutions, in other words they permit risk to be spread.

The move away from a monoculture has recently gained considerable momentum. For example the Australian Federal government is currently proposing legislation that open source solutions must be considered in government procurement cycles [22]. On the other hand Korean and Japanese governments are actively looking for open solutions to Microsoft {Myung, 2003 #29}. It is planned that by 2007, 30% of Koreas government infrastructure (enterprise and desktop) will be running Linux [23]. Statements suggest that a major concern of government departments is reducing the total cost of ownership and locking down into a particular technology which may make them more susceptible to the vendor/ attacks. 34% and 47% of respondents from the OpenForum Europe Survey [25] cited these are the two core reasons for using open source. Specific case studies such as the Gundagai Shire Council [17] finance system project saw a legacy system be moved to open source technologies despite two other options involving cost Commercial Unix and Windows Solutions. This case study illustrates the success of open source by maintaining interoperability whilst reducing over total cost of ownership.

5. Open Source Strengths

A major strength of the open source movement is the licensing arrangements, which provoke and actively promote community involvement and recognition. The GNU Public License (GPL), Berkley Software Distribution License and Mozilla Public License all aim to varying degrees to allow members of the open source community to share in a open fashion the fruits of their labour whilst protecting their works [27]. These agreements to varying levels control how source code can be used by third parties. These licensing arrangements differ from the current model offered by a number of vendors. For example Microsoft tries to mimic the GPL through its Microsoft Shared Source license [28] that provides a select 'partners' conditional access to selective components of source code which are significant to the work they are undertaking (Microsoft, 2004). The differing license models, particularly the restrictive nature of the Microsoft licensing arrangement is the basis of this monoculture. This attitude is very different to those organizations that Zeltin [29] identifies as taking a further step. "Some companies and government organizations are taking their commitment to open source a step further by actively participating in the open source community."[29] It can be seen that open source is more dynamic then closed source partially due to the licensing arrangements and terms.

Many however see open source software as a threat to the software industry, particularly in reference to livelihoods and intellectual property. Typically vendors develop software that suits a specific agenda or is economically viable [30] thus more often the not leaving consumers in the dark. In recent press articles it is stated that "Microsoft is making an 80% mark up on Windows and Office."[31] The attitude of large vendors is one that describes software as being a privilege however it is becoming clear that software is becoming a commodity [31]. There is an expectation today that software be made accessible at reasonable cost. Open source meets this challenge whilst the propriety models with its licensing arrangements fail miserably.

The ability for the community to shape software is a significant benefit of the open source movement yet seen as a major threat to its uptake partially due to perceived lack of direction/ structure [32]. One of the most significant factors contributing to the lack of open source uptake was ‘direction and uncertain futures’ {Griffiths, 1999 #37}. That said the authors believe that uncertainty provides consumers with a technology that can mature with them and is not bound to the roadmaps/ agenda of a specific vendor. This can be best illustrated with international efforts to localise software suites such as Open Office, GNOME and KDE which are central to any Linux distribution yet seen as unpractical by large vendors such as Microsoft [30]. Members of the community from across the world have participated in making open source software solutions such as these localised so that consumers are not left in the dark.

Users from around the world are discovering the benefits of open source over the traditional proprietary models. In the past vendors such as Microsoft, Apple and Sun and have written software solutions for consumers and more often then not consumers have had no say in the process. It has only been in recent years that such vendors have begun to open their minds to the Open Source idea by either embracing technologies which have resulted from Open Source Projects or changing there stance on community involvement. This can be best illustrated with changes at Sun Microsystems with the development of the Sun One (Open Network Environment) suite which features a number of projects from the open source community [33]. Other examples of this change can be seen at IBM with their embracing of the Linux platform [34] Apple’s Open Source Licensing Agreement enabling users to contribute to a commercial industry strength operating system [35] and specific niche projects from Sun Microsystems such as the Open Sun Grid Engine [36].

6. Conclusions

The inherent dynamics of open source software development and perceived lack of development direction is often seen as one of the major drawbacks to business uptake. However as shown, many of the perceived lacks and uncertainties of Open Source software compared to proprietary software are in fact prime reasons why businesses should be including OSS in their application and systems mix. The heterogeneous nature of OSS can help to reduce the impact of vulnerabilities and security breaches.

It can be argued that these changes in ‘computing culture’ are seeing a new era in software development occur. Vendors now appreciate the dynamic nature of open source and can see that heterogeneity in platforms and software solutions enhances the computing/ consumer experience. Thus the principles underlying Open Source development provide a higher quality software experience on many levels.

References

[1] CERT/CC, CERT/CC, *CERT/CC Statistics 1988-2003*, 12 March 2004, [HTML], http://www.cert.org/stats/cert_stats.html

[2] ISO, “ISO 8402,” International Standards Organisation 1994.

[3] J. E. Gaffney, “Metrics in software quality assurance,” presented at Proceedings of the ACM '81 conference, 1981.

[4] B. W. Boehm, J. R. Brown, and M. Lipow, “Quantitative evaluation of software quality,” presented at 2nd international conference on Software engineering, San Francisco, California, United States, 1976.

[5] F. O'Brien, “Business metrics for software-based systems,” SEA Software Australia, pp. 17- 20, 2001.

[6] D. E. Harter and S. A. Slaughter, “Process maturity and software quality: a field study,” presented at twenty first international conference on Information systems, Brisbane, Queensland, Australia, 2000.

[7] G. Denaro, S. Morasca, and M. P. e, “Deriving Models of Software Fault-Proneness,” presented at SEKE '02, Ischia,Italy, 2002.

[8] R. Dawson, *Living Networks: Leading your Company, Customers and Partners in the Hyper-Connected Economy*. Upper Saddle River, New Jersey, USA: Finacial Times, Prentice Hall, 2003.

[9] P. G. Neumann, “Risks to the public in computers and related systems,” *ACM SIGSOFT Software Engineering Notes archive*, vol. 26, pp. 6 - 15, 2001.

[10] OpenSource, *The Open Source Definition*, 10 March 2004, http://www.opensource.org/docs/definition.php

[11] J. Dinkelacker, P. K. Garg, R. Miller, and D. Nelson, “Progressive Open Source,” presented at International Conference on Software Engineering, Orlando, Florida, USA, 2002.

[12] M. Wheatley, CIO Magazine, *The Myths of Open Source*, 12 March 2004, [online], http://www.cio.com/archive/030104/open.html

[13] M. J. Karels, “Commercializing Open Source Software,” Queue,, vol. 1, pp. 46-55, 2003.

[14] E. S. Raymond, *The Art of Unix Programming*, 1st ed. Boston, USA: Addison Wesley, 2003.

[15] U. Tung, "Linux and the Open Source Movement: A Sun Microsystems Perspective," presented at The 11th Annual System Administrator Guild of Australia Annual Conference, Hobart, Tasmania Australia, 2003.

[16] M. Widenius, "Workshop Notes: MySQL Administration for Beginners," presented at The 9th Annual System Administrator Guild of Australia Annual Conference, Adelaide, South Australia, Australia, 2001.

[17] C. Levanes, "Lowering the total cost of ownership though open Source Software," presented at Australian Unix Users Group Conference - Always On, Everywhere, Sydney, Australia, 2001.

[18] A. Tridgell, "Towards Full NTFS Semantics in Samba," presented at Linux.Conf.Au, Perth, Australia, 2003.

[19] Sohphos Systems, *Blaster Worm Exploits Microsoft Security home and targets critical update website, Sophos Antivirus warns*, 12 August 2003, [online], http://www.sophos.com/virusinfo/articles/blaster.html

[20] CERT/CC, *Cert Incident Note IN-2003-03 W32/Sobig.F Worm*, 22 August 2003, [online], http://www.cert.org/incident_notes/IN-2003-03.html

[21] P. Rooney, *CCIA Blasts Government Dependancy on Microsoft Windows, Security Pipleline Newsletter*, 25 September 2003, [online], http://www.securitypipeline.com/showArticle.jhtml?articleID=15200503

[22] F. Foo, *Analyst blasts open-source legislation*, November 2003, [online], http://www.zdnet.com.au/news/business/0,39023166,20278573,00.htm

[23] S. E. Myung, *Korea Lauches a Switch to Open Source*, 1 October 2003, http://news.com.com/2100-7344-5084811.html

[24] Security Focus, *Security Focus, Vulnerability Statistics 1998 - 2001*, 10 March 2004, [online], http://www.securityfocus.com/vulns/stats.shtml

[25] OpenForum Europe, *Open Source - Coming of Age*, July 2003, [online], http://www.openforumeurope.org/publications/open_source_coming_of_age/

[26] K. Lakhani and J. Bates, "Boston Consulting Group Hacker Survey," presented at Linux.Conf.Au, Perth, Australia, 2003.

[27] J. Malcolm, "Problems in Open Source Licensing," presented at Linux.conf.au, Perth, 2003.

[28] R. Hart, "Through the Looking Glass: Open Source, Microsoft and public perceptions," presented at Australian Unix Users Group Conference - Always On and Everywhere, Sydney, Australia, 2001.

[29] M. Zetlin, Computerworld, *In the Linux Loop*, April 2003, [online], http://www.computerworld.com/softwaretopics/os/story/0,10801,80053,00.html

[30] MONITOR, Economist.com, *Open Source Local Hero's*, http://www.economist.com/science/tq/displaystory.cfm?story_id=2246308

[31] S. Beer, "A new door to open," in Sydney Morning Herald Australia, 2004.

[32] D. Griffiths, "Business Attitudes to Open Source," presented at Australian Unix Users Group Conference - Open Source, Melbourne, Australia, 1999.

[33] Sun Microsystems, Sun Java Desktop System, 9 March 2004, [online], http://wwws.sun.com/software/javadesktopsystem/index.html

[34] C. Jay, "Australia to be a Linux Battleground," in Australian Financial Review, 2003, pp. 61.

[35] Apple Computer Inc, *Open at the Source*, 7 March 2004, [online], http://www.apple.com/opensource/

[36] Sun Microsystems, *Open For Innovation*, [online], http://wwws.sun.com/software/cover/2001-0724/