VOL: 5, ISSUE: 2
February/2019
https://ijbassnet.com/
E-ISSN: 2469-6501

System Complexity and the Probability of Flaws in the Post Sarbanes Oxley Era

Jorge Romero, Ph.D.
Associate Professor
College of Business and Economics
Towson University
E-mail:jnromero@towson.edu
USA

Abstract

Information is a key asset for accounting processes, auditors, and businesses in general it is essential to understand accounting information systems infrastructures and the programs that access our valuable data. Systems that we believe are secure and protecting our data may be opening themselves to various risks caused by the very components that we have added for security purposes. The thousands of lines of code and additional processes we add to create a more secure environment may be leading us down a road of security vulnerabilities because of the added complexity to these systems. Accountants, auditors and business professionals in general should take steps to ensure that their data is secure; not only from the typical computer security risks such as hackers, viruses, natural disasters and theft, but from the risks arising from the system itself where the data resides. With this study, I hope to encourage a more strenuous approach to information security through the prospect of also looking at the implementation of components and complexity as causing flaws in accounting information system applications.

Keywords: Accounting Information Systems, SOX, Sarbanes-Oxley Act, Complexity

1. Introduction

As technology has infused itself with most business transactions, we have become ever dependant on complex systems of software to handle our everyday activities. With routine interactions with our computers, databases and Accounting Information Systems we often do not stop and consider what lies beneath the surface of this indispensable programs or what not knowing could mean for your company. Since information is a key asset for accounting processes, auditors, and businesses, in general, it is essential to understand accounting information systems infrastructures and the programs that access our valuable data. This is of growing importance especially after the Sarbanes-Oxley (SOX) 2002 Act was passed; maintaining data integrity and software security needs to be a top priority to ensure that data is accurate to stay in compliance with new laws. We do not know much about SOX'spost affects on software complexity or security since companies were forced to be in the compliance year 2004. Even though studies that investigate the effects of SOX on material weaknesses and the strength of internal controls, they still lack sufficient data that can be applied to business complexity, specifically accounting information systems(Doyle et al, 2007; Krishnan & Visvanathan, 2005). These studies also utilize the limited available data in the post-SOX era of business practices. The long-term effects of the Sarbanes-Oxley Act are still not known and little evidence exists relating to the overall quality of internal controls under SOX (Doyle et al, 2007). I see SOX as adding complexity to business practices overall, thus compounding the complexity of IT structures and software code in order for the business to stay in compliance and limit control deficiencies.

VOL: 5, ISSUE: 2
February/2019
https://ijbassnet.com/
E-ISSN: 2469-6501

2. The Sarbanes-Oxley Act

The addition of the Sarbanes-Oxley Act into business practices has caused many unforeseen consequences adding complexity to numerous management behaviors (McCarthy, 2004). Many changes were necessary following the uncovering of business scandals in the corporate world, such as the discovery of Enron and other appalling business practices.Through the regulations specified by SOX, strict penalties for managerial misconduct and unscrupulous business practices are set in place to discourage unethical or unlawful conduct and misrepresentation (Zhang, 2002). To ensure that companies are following governmental guidelines, related audit documentation must be kept and secured for a period of time which impacts the technology requirements of businesses. These changes in business practices not only influence one's own company and its practices but have affected smaller firms and clients indirectly as well (McCarthy, 2004).

Systems that we believe are secure and protecting our data may be opening themselves to various risks caused by the very components that we have added for security purposes. The thousands of lines of code and additional processes we add to create a more secure environment may be leading us down a road of security vulnerabilities because of the added complexity to these systems. Like "adding the last straw to the horse's back", adding an extra bit of code to an already complex system often can add that extra fault, inadvertently creating an unstable system (Lehman&Belady, 1985). In the real-world systems are not static, but require maintenance and updates as business requirements evolve over time and to keep up with new accounting practices. Even the most precise implementation of a computer security policy for a complex system is bound to contain security flaws as changes are made to the system to accommodate these changing requirements (Hofmeyret al, 1998).

3. The Role of Accounting Professionals

Accountants, auditors and business professionals, in general, should take steps to ensure that their data is secure; not only from the typical computer security risks such as hackers, viruses, natural disasters, and theft but from the risks arising from the system itself where the data resides. Staying in compliance with Sarbanes-Oxley, for instance, is just one of many reasons to keep your data in check. It is not just a matter of your network administrator installing an extra firewall or a simple case deleting potentially harmful emails from your inbox…these internal threats are unpredictable like a time bomb that may or may not explode into a frenzy of data loss and corruption. One would believe that the seriousness of these threats alone would prompt IT, practitioners, to pay close attention to methods to protect computer systems, but many organizations and business still lack adequate protection or remain completely unprotected in key areas (Straub &Welke, 1998). Management has a responsibility to oversee that proper security precautions are in place and to investigate any potential problems in software that may lead to weak spots in the company's security infrastructure. Even though models exist that show proper levels of security investments, management in many instances lacks the necessary resources to protect company assets (Gordon & Loeb, 2002). Management also may not have a realistic understanding of the potential security threats or the related risks (Tudor, 2000). Even if management does understand what the risks are, they may not be aware of the appropriate actions to reduce systems risk (Straub &Welke, 1998). It is not sufficient to wait until the threat has been uncovered "naturally" by its inexplicable appearance during a critical moment (as it most often seems to happen). Understanding the nature of complex systems can help management better understand and prepare for the problems they are facing when it comes to software. Sarbanes-Oxley has essentially helped professionals understand the importance of information security and its greater implications for businesses by emphasizing the importance of safeguarding data, but at the same time added more complication (Stults, 2004).

4. Electronic Storage of Documents

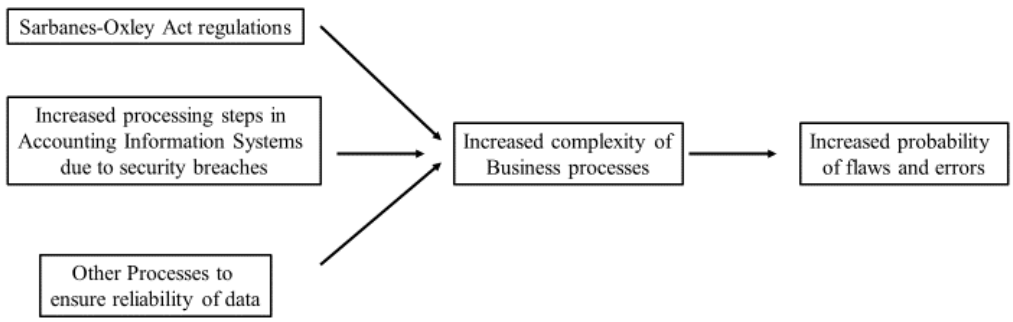
With Sarbanes-Oxley in the equation of business practices, the concern of accurately preserving documents and related audit information is a security concern. Storage is also important, but maintaining and preserving these documents electronically in a paperless office should be a top priority. Since SOX requires that documents be

VOL: 5, ISSUE: 2
February/2019
https://ijbassnet.com/
E-ISSN: 2469-6501

retained, any unprotected documents should be archived and protected. Even with the promises of software vendors and their assurance of their quality software, can we be sure that data will not become lost or corrupted, caused by the system that was designed to protect the data itself? No matter the type of system used to preserve important financial documents, it should do so in a manner that maintains file integrity and quality. Even though we cannot have a system that is absolutely secure we can devise relatively formal methods to effectively improve planning for security (Straub &Welke, 1998). We also should consider that many large systems work through the interaction of several layers of interconnected systems and devices. For instance, there may be some e-commerce website that functions through its interaction with a database that contains product inventory and other relevant data, a web server that contains the code that drives the site and then a connection to an outside system that processes credit card payments for customer purchases. These systems must work together in a cohesive fashion in order to create the e-commerce website; it is not a single entity in its own right. Complexity can arise from these types of interactions because they are essentially systems composed of other systems built from various complex software and telecommunications media which also uses complex software (McDermid, 2000).

The probability that a company's information infrastructure contains inherent flaws and is vulnerable to various levels of attack is a real concern. It has even been suggested that the foundation of these systems, i.e. the hardware itself, introduces potential security threats by compounding the issue of complexity at the microprocessor level potentially introducing mathematical and computational errors (Markoff, 2007). Add yet on top of this, complex applications consisting of thousands of lines of code often with deficient, obscure or out of date documentation (Forward & Lethbridge, 2002; Smith et al, 2001). As systems become large and more complex it seems expected that flaws will emerge. Even with a relatively stable program, complexity in an application can cause unpredictable behaviors when unplanned interactions take place or changes occur in its operating environment (Gribble, 2001). We simply cannot predict how all the components of a complex system will operate all the time. This unpredictability and uncertainty should raise questions over the emergence of security risks in the systems we use every day. Layers of abstraction serve to hide implementation details of underlying software components but may be burying flaws inadvertently, which later are exemplified by the interaction of these objects. Unforeseen flaws in this sense emerge from the interaction of the simple pieces introducing higher level complexity issues and security vulnerabilities.

Figure 1: The Link between System Complexity and Probability of Flaws



VOL: 5, ISSUE: 2
February/2019
https://ijbassnet.com/
E-ISSN: 2469-6501

5. Reliability of Data

Due to legislation and the value of information that is being stored in large systems and accounting databases, the reliability of software in the financial realm is of great importance. A lack of quality elicitation of the software requirements in the beginning phase of the design process can have long-lasting and devastating results leading to a "variety of deficiencies" which leads to "incompleteness, contradictions and ambiguities" (Lamsweerde, 2000). Who can say that one person or a team of people have the correct idea of the overall program design in order to properly convey the correct needed operations and scalability requirements before a firm's application is developed? To develop a dependable system, a significant amount of time and effort is required to properly test and design a program (Ostrand et al, 2005). Many off the shelf software components are known to be versatile in the tasks they can complete but have added to the large amounts of functionally that is not needed in many cases adding complexity (McDermid,2000). With all the many components involved with these systems, we should also look at each component and its function. There is inherent complexity it the interface between hardware and software components which are needed for the implementation of applications (Courtois &Parnas, 1993). We cannot be sure if these components themselves are completely trustworthy or their effect on higher level components. Often companies develop independent modules without divulging the source code or internal functionalities. Requirements documentation describes components of systems as a "black-box" leaving out internal workings only describing the variables and relationships (Courtois &Parnas, 1993). Without knowing how all of these components work, we cannot with certainly know how these components will work under all circumstances or how stable they will remain over time. Complex systems with myriads of coupled pieces can be affected by many things, which can lead to a "butterfly effect" causing large or unknown behavioral output (Gribble, 2001).

Imagine that your company does, in fact, have a relatively stable accounting information system and that your IT department and management at your firm keep up with all the new recommended upgrades and security measures. Your firm even makes regular backups of all transactions, financial data and important documents in case of a catastrophic hardware failure. Even with this sense of security, adding the latest security patch can conflict with other application programs or previously installed components that worked before. What once properly performed now has extra added code that now may have just introduced an unpredictable error or behavior. The same can be said if we install new hardware into the mix of an already complicated system. The complexity of systems is growing as we scale software and other devices to increase their capabilities, but these systems are becoming more difficult to maintain (Ho, 2002). If the original design of the system was designed for a particular input device, upgrades have now allowed the system to work with other technologies; the added components of the software may compromise the integrity of the original system architecture. Even robust and optimized large scale systems can potentially be made "fragile" when operating under conditions in which it was not originally designed (Ho, 2002).

6. Conclusions

The issue of complexity is not limited to merely the software itself but also through its interactions with external sources (McDermind, 2000). An important and very unpredictable element in the overall security of the system comes from the human element and how we interact with these complex systems. We must ensure that user access is restricted to appropriate levels so unauthorized employees cannot make changes to financial records or other documents. Another issue is with the implementation of the security features themselves and how we use these features. Adding security inherently adds complexity as it adds more code to the system. Even though we can't get around not having security mechanisms in place, as with any design it's important to eliminate complexity as much as possible, to hinder potential stability problems or added bugs (Berson, 2005). Complexity is of itself a complex issue that can stem from related elements of a systems scale, diversity of components or its connectivity to other elements in terms of relationships or interactions (McDermid, 2000).

VOL: 5, ISSUE: 2
February/2019
https://ijbassnet.com/
E-ISSN: 2469-6501

The growing number of complex and large software systems introduces internal complexity resulting in inherit security flaws. Through the addition of security and secure devices does not facilitate a more secure system but compounds the issue of complexity in subsequent layers. These issues along with the use of less protected hardware devices leave open blind spots in otherwise, thought to be secure systems. With this study, I hope to encourage a more strenuous approach to information security through the prospect of also looking at the implementation of components and complexity as causing flaws in accounting information system applications. Exploits often start from small scale "cracks" within a security infrastructure which has been exemplified by the introduction of legislation such as SOX which has added more complexity to business practice which is reflected by the need for larger systems to stay in compliance.

References

Berson, J. (2005) ZoneAlarm: Creating Usable Security Products for Consumers. In Security and Usability. Cranor, L & Garfinkel, S (Eds.)

Courtois, P.,&Parnas, D. (1993) Documentation for Safety Critical Software. Proceedings of the 15th international conference on Software Engineering. IEEE.

Doyle, J. Ge, W., &McVay, S. (2007) Determinants of Weakness in internal control over financial reporting. Journal of Accounting and Economics.doi:10.1016/j.jacceco.2006.10.003

Forward, A.,& Lethbridge, T. (2002) The Relevance of Software Documentation, Tools and Technologies: A Survey. DocEng '02 McLean, Virginia ACM.

Gordon, L.,& Loeb, M. (2002) The Economics of Information Security Investment. ACM Transactions on Information and System Security, Vol. 5. No 4. pp 438-457.

Gribble, S. (2001). Robustness in Complex Systems. Proceedings of the eights workshop on Hot Topics in Operating Systems. (HotOS-VIII '01) IEEE

Krishnan, G.,& Visvanathan, G. (2005) Reporting Internal Control Deficiencies in the Post-Sarbanes Oxley Era: The Role of Auditors and Corporate Governance. Available at SSRN: http://ssrn.com/abstract=646925

Lamsweerde, A. (2000) Requirements Engineering in the Year 00: A research Perspective. 22nd International Conference on Software Engineering (ICSE '00) p.5

Markoff, J. (2007) Adding Math to List of Security Threats. The New York Times, November.

McCarthy, Ed. (2004) Tips for the Sarbanes-Oxley Learning Curve. Journal of Accountancy.

McDermid, J. (2000) Complexity, Causes and Control. Proceedings of the 6th IEEE International Conference on Complex Computer Systems. (ICECCS '00)

Smith, D., Thomas, B.,& Tilley, S. (2001) Documentation for Software Engineers: What is Needed to Aid System Understanding? SIGDOC 2001 Santé Fe, New Mexico ACM.

Stults, G. (2004) An Overview of Sarbanes-Oxley for the Information Security Professional. SANS Institute GSEC Practical Assignment Version 1.4b

Tudor, J. (2000) Information Security Architecture, An Integrated Approach to Security in the Organization. Auerbach Publications

VOL: 5, ISSUE: 2
February/2019
https://ijbassnet.com/
E-ISSN: 2469-6501

Ho,Y., Zhao, Q.,&Pepyne, D. (2002) The No Free Lunch Theorems: Complexity and Security. IEEE Transactions on Automatic Control, Vol. 48, No 5. May 2003.

Hofmeyr, S., Forrest, S.,&Somayjai A. (1998) Intrusion Detection Using Sequences of System Calls. Journal of Computer Security Vol 6. Issue 3 pp. 151-180 ACM

Lehman M.M.,&Belady, L.A. (1985) Program Evolution: Processes of Software Change.London: Academic Press.

Ostrand,T.,Weyuker, E.,& Bell, R. (2005) Predicting the Location and Number of Faults in Large Software Systems. IEEE Transactions on Software Engineering (Vol. 31, No. 4) pp. 340-355

Straub, D.,&Welke, R. (1998) Coping With Systems Risk: Security Planning Models for Management Decision Making. MIS Quarterly, Vol 22, No. 4, pp 441-469

McDermid, J. (2000) Complexity: Concept, Causes and Control. Proceedings of the 6th IEEE International Conference on Complex Computer Systems (ICECCS)

Zhang, I. (2002) Economic consequences of the Sarbanes-Oxley Act of 2002. Journal of Accounting and Economics.