SYSTEMS THEORY MODEL FOR INFORMATION SECURITY

Wm. Arthur Conklin
University of Houston
waconklin@uh.edu

Glenn Dietrich
The University of Texas at San Antonio
glenn.dietrich@utsa.edu

Abstract

Architecting security solutions for today's diverse computer systems is a challenge. The modern business environment is comprised of many different applications, e-mail, databases, e-commerce, and more. Each of these has its own threat profile and associated business risk. The complexity of the computing environment extends to the design of security solutions. Current methodologies for designing security systems include piecemeal designs and patchwork systems comprised of multiple point solutions. As the complexity of the business driven systems increase, these methods are being strained to keep up with security requirements. Systems science provides information on how complex systems interact with their environment, and this guidance can be applied to designing security architectures. Analysis and design of security systems using systems theory provides a new path to reduce the complexity.

1. Introduction

As modern business organizations increase their reliance on information systems to meet the challenges of the competitive marketplace, the importance of security becomes an important issue. Designing security solutions in today's diverse computing and growing threat environment is a challenging and complex task. This research paper explores the problem of architecting a system level security response and the implications and limitations that arise from systems theory.

Security can act as a control mechanism in a system. Security mechanisms are designed to allow desired events and block undesired ones [9, 20]. When computer systems were simple and single purpose in nature, the design of control mechanisms to provide for security was a relatively simple and well understood task [20]. As the use of computer systems in business expanded in the past several decades, so have the number of application, including e-mail, office automation programs, accounting programs, and databases. The Internet expanded the scope of the problem further, with the addition of e-commerce, instant messaging, web sites, web services and more. During the period of expansion in function came the era of system consolidation and system integration. Major vendors began offering suites of solutions to various business problems. This combination of application integration, server consolidation and enhanced business services was designed to increase business efficiency, productivity and business responsiveness. The simplifications offered by these changes did not extend to security functionality.

During the same period, the threat to information systems has changed dramatically, from an age of simple network attacks designed to demonstrate hacker prowess, to today's integrated and sophisticated botnet attacks designed to specifically defraud companies and users [15]. Increasing levels of attacks have lead to more frequent security breaches, and the losses associated with them has resulted in a heightened level of importance to company executives. Recent surveys have indicated that the majority of organizations consider information security as highly important to achieving their overall business objectives [11, 18]. Information security vendors and researchers have responded to the changes through the development of a wide range of technology based solutions.

Technology based solutions and research have been based on point solutions, with differing answers for access control [1, 5], encryption [21], and protective devices such as firewalls [17] and intrusion detection systems [6, 10]. The myriad of point solutions have lead security professionals to navigate a complex minefield of options and capabilities. Guides to deployment strategy exist through a series of principles, such as defense in depth [9]. The result has been a patchwork and piecemeal

1530-1605/08 $25.00 © 2008 IEEE

deployment of a diverse set of technological solutions in an attempt to secure the enterprise. The result, when measured against the growing list of security incidents is proving to be less than desired. An examination of the problem through the lens of systems theory provides some interesting models of understanding of this complex problem.

Systems theory has been applied to many areas of study [7], and systems engineering approaches have guided many firms in their security offerings. A review of the security literature has not revealed significant academic analysis of the use of systems theory in the analysis of information security. This paper will explore the application of a specific aspect of system theory, the law of requisite variety, to the problem of security controls implemented as regulator based functions in a system.

The use of systems theory in this way will provide information to designers, developers, and security professionals in their attempt to understand the limitations and basis for limitations with respect to this critical functional aspect of their systems. This is the application of what has been described as 'hard' systems theory as opposed to 'soft' methodologies by Checkland and others. The application of Soft Systems Methodology (SSM) is also applicable in differing manners but is outside the scope of this paper.

2. Security Issues

The importance of security is echoed by reports from the Computer Security Institute and FBI [12], and industry groups such as CompTIA [8], and RSA Conferences [25]. But as the importance is placed at the entire system or enterprise level, the solutions that are employed are at a single specific system level. Different security technologies have been developed to combat specific levels of security threats. These threats can be divided into a range of different categories, such as, network attacks, operating system attacks, social (people) attacks, and application level attacks.

2.1. Network Attacks

Network level attacks are attacks designed to attack the network infrastructure. Typical network attacks include items such as Denial of Service (DOS), distributed denial of service (DDOS), and man-in-the-middle attacks (MITM). These attacks are designed to disrupt specific actions of the network, allowing an attacker to influence how a network behaves during specific events. Domain name system attacks, ARP poisoning and other attacks are designed to influence how a network communicates addresses and delivers information. When examining the computer system in terms of the OSI model [9], these attacks occur at the lower levels. Some of these attack mechanisms are perpetuated with single packets across the network. Protection and defensive efforts against these attacks is accomplished by devices that focus their efforts on the first three levels of the OSI model.

2.2. Operating System Attacks

Another level of attack is against the specific operating system employed in a system. Operating system attacks take advantage of software flaws in these complex programs. The attack is typically designed to enable an attacker to gain a specific level of user access, frequently root or admin access, when such access is not appropriate. These access based attacks can also be designed to change the functionality of the operating system, enabling unauthorized parties access to information that they should not have access to. Key loggers and ad-ware are examples of common threats in this classification.

2.3. Application Level Attacks

Computer systems exist, not for the network, or for the operating system, but for the specific applications being employed by the business. Application level attacks are attacks against specific applications being employed by the business. Similar to operating system attacks, these attacks are typically against software flaws or misconfiguration in deployment. The objective of these attacks is again the same as in operating system attacks, to gain access to information which should be protected. Once access is obtained, information can be stolen, changed or deleted. E-mail virus attacks designed to clog e-mail servers, SQL injection attacks designed to change data in databases, and the use of buffer overflows to inject code into a system are some examples of these types of attacks.

2.4. Social (People) Attacks

Different than all the previous types of attacks are attacks aimed at the end user. People have often been characterized as the weakest link in modern computer systems [2]. End users can be tricked into revealing information through attacks, such as phishing , human error [16], or the end user may be the attacker as in the case of insider attacks [24]. People operate outside the typical control mechanism of many technology systems, being controlled by corporate policies and procedures [19]. It has been stated that there is no firewall for the end user, and attackers take advantage of this weakness.

2.5. Mixed and Blended Attacks

Many modern security incidents occur not because of the failure of a single level of control or technological mechanism, but because of multiple failures. Firewalls are designed to prevent specific types of packets from crossing specific network boundaries, yet need to allow other packets to flow freely. Bypassing a firewall is as easy as only sending traffic across the device that conforms to specific rules, appearing as normal traffic. Once inside, the attacker can then change tactics and attack a different level of protection, such as the operating system or applications. Viruses, worms, malware and spy-ware payloads are designed to appear as ordinary business traffic, making specific detection and filtering more difficult. But the camouflage used to prevent technological discovery is only the first level of camouflage, the package also must appear proper to the end user to invoke a desired end user interaction resulting in the release of attack against the system.

3. System Theory

A system is defined as a collection of entities that act together to perform a specific purpose [3]. A system is separated from its environment by a boundary, which separates what is in the system and that that is not. Entry to the system, across the boundary is done through the inputs of the system. The other crossing of the boundary is performed by outputs from the system, which provide information back to the environment. A system may contain a regulator component, designed to manage the control of the output, based on the input of a system. The regulator component operates against information inside the boundary of the system only. One manner of describing a system is to define the states that the system can exist in, and the method of transition between these states. The linkage between the system and controller has been explored by many systems scientists. The law of requisite variety defines the minimum number of states necessary for a controller to control a system of a given number of states [3]. This minimum number is at least as large as the number of states that can exist in the system. This implies that the number of states in a system can be calculated and defined. The regulator component acts through a feedback mechanism to achieve its goal of managing the states in a system within the desired limits of control. For the regulator to act, it must have connections to the inputs and outputs of a system. The regulator cannot regulate for states outside the bounds of the system, in fact it can be seen as model of the system within the system boundary [4].

A simple system is illustrated in Figure 1. This system has two inputs, a single system element that produces an output based on the two inputs, and a single regulator element. The system boundary line separates the system from the environment. A complex system can be designed from a collection of simpler systems. The individual, simple systems are connected together into a larger, more complex system through a series of direct, circular, and time delayed interconnections. Subsystems can be nested within each other, or in an overlapping fashion as necessary to decompose a complex system into manageable pieces. The actual decomposition of a complex system into a set of subsystems can be done in many ways. The proper method of decomposition is determined by the reason for decomposition. Figure 2 depicts some examples of complex system decomposition. When analyzing the actions and control of a system, the boundary and regulator determine the scope and range of control of the delineated system.

The actions of the complex system as a whole can exhibit properties that are not directly attributable to individual system elements. These properties are referred to as emergent properties. Security is frequently an emergent property of a system [23], not specifically pegged to a single component, but one where several components interact to produce the desired result.

Examining the entirety of Figure 2 as a system, then the regulator portion can be contained in virtually any combination of the elements. As a system becomes more complex, the number of states increases and the size of the regulator function must correspondingly increase to cover all the possible system states. This becomes particularly problematic in several cases common to information security systems, namely the changing nature of the environment and the issues of emergent and unknown states. Information security involves a continually changing threat landscape, which is represented in the environment aspect of the model. Unknown interactions between components are also common in IT, and a mix of old and new technologies has led to many vulnerabilities such as the Windows Metafile issue of 2005.

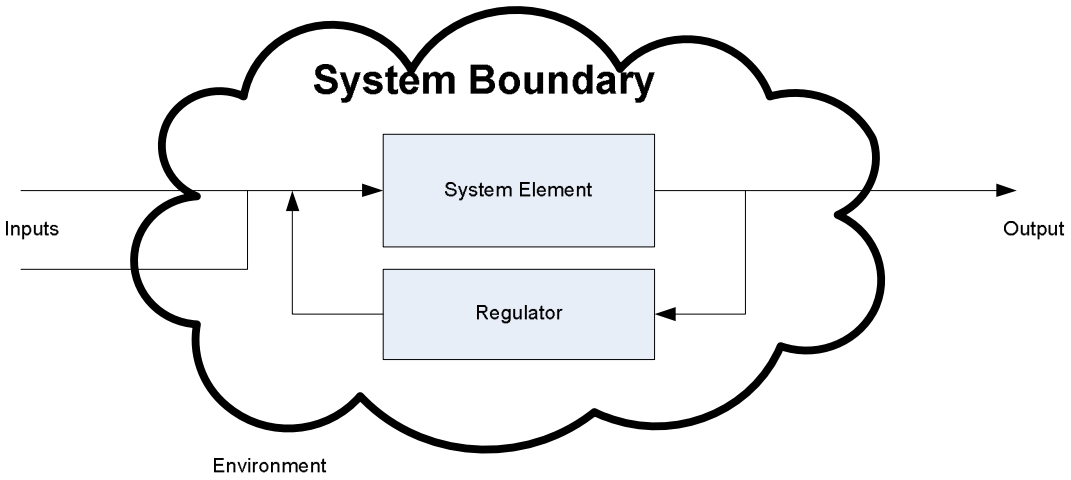

Figure 1. A Simple System

4. System Design

The overall design of a complex system is the venue of the system architect/engineer. Systems engineering is the process of designing and developing multiple interconnected components in such a way that they function efficiently together to perform specific tasks which meet specific needs in an organization. A basic tenet of system engineering is the concept that the whole can do more than the sum of the parts. This is seen in many IT enterprises, as IT departments connect business units with database systems, email and file sharing servers and systems as well as web based e-commerce and web services. Each of these named systems is a complex system in its own right, and can be decomposed into smaller, simpler systems. The anti-virus hooks in an email server, that are designed to intercept email attachments and scan for malware are examples of the interconnected nature of these systems. To design security into a system requires a system level of thinking, for the design must take into account the interaction between components of the system, and the resulting emergent properties or lack thereof, when attempting to achieve specific levels of security [26].

One of the philosophies behind information security is that there is no such thing as absolute security. With this in mind, designers incorporate a series of controls, acting in series to provide a layered defense, or defense in depth. This idea is correct and makes sense, yet it is not properly implemented in many occasions. It is common for a modern enterprise to have firewalls, passwords, access control lists and patch programs - which together, they describe as defense in depth. Yet as a systems theory based analysis will show, some of these mechanisms are totally independent and will not improve detection and prevention when employed in serial fashion.

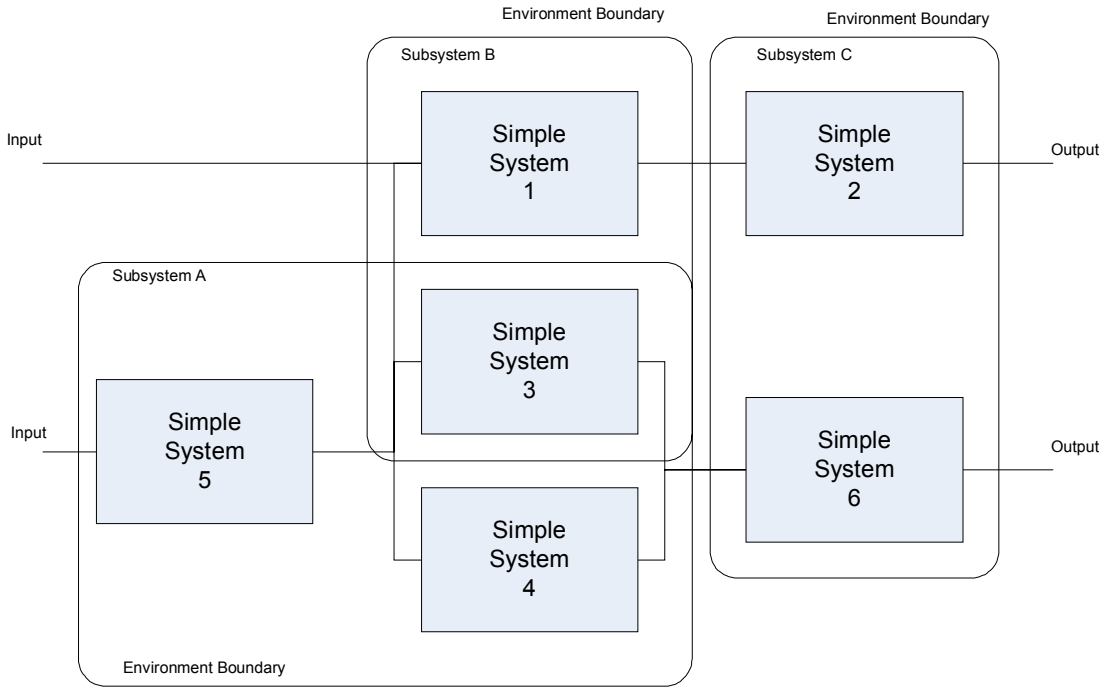

Figure 2. A Complex System

5. System Theory Meets Security

Looking at the security puzzle with the lens of systems theory sheds specific light on a different level of aspects. One of the principles of security is defense in depth, the use of multiple defenses to achieve higher levels of protection [22]. This concept is really only applicable if each level has an opportunity to be effective against a specific threat. Firewalls may be one line of protection in a system, but they offer no real protection against application level attacks, such as SQL Injections. Application specific firewalls have been designed to combat specific application threats, such as database threats, xml based threats and email threats, yet these firewalls have no effect against network specific attacks. Perimeter based measures, although effective for some threats, are not effective for other threats.

A systematic examination of different types of attacks and how they interact with the overall IT system provides insight into the visibility of specific items to a regulator or control aspect of the system. As the regulator can only monitor items visible within the boundaries of a system then there are signals that can transit a system without detection or control by the regulator. Take the example of a network appliance designed to monitor network traffic for specific forms of malware. The simplest unit of information transfer is a single packet, and a network level device can examine each and every packet as it crosses the boundary into and out of the system. As such, single packet threats can be observed by the regulator and hence controlled. The level of complexity of the regulator is equivalent to the number of states that can be expressed in the system under control; hence it is bounded by the number of signatures that are being scanned for, and is thus a finite number. Expand this to multiple packet attacks and the problem grows in complexity. In addition to monitoring events over multiple packets and the work of reassembly, multiple concurrent conversations also may need to be monitored. A network level appliance that desires to scan all incoming email for malware threats has to first group packets, by conversation, building each separate email message and attachment in whole. These complete items can then cross the boundary and

be observed and controlled by an appropriate regulator. In small systems, this may not be a large endeavor, but as the number of email conversations increase, the level of complexity increases significantly. The law of requisite variety states that in this case, the "system, including regulator" must be of equal size as all states of the system, which is scaled by the number of emails and sizes of attachments. In a single packet analyzer, the scope of the environmental boundary for the control system is a single packet. When multiple packets are needed for analysis, the boundary changes to that of completed units of work, whether this be just a few, or thousands of packets.

For a network based Intrusion Detection System, the regulator function would be to differentiate authorized and non-authorized traffic. Again, the scope of the regulator function is dependent upon the number of states in the system under control. If the regulation function is going to operate off of a simple black list or white list of IP addresses, then the control system boundary can be set at a single packet. Each packet contains the source and destination IP addresses, so a boundary constructed at the single packet arrival point is sufficient. If multiple packets are necessary for determination of control functions, then the boundary needs to be moved to a system that has accumulated the sufficient number of packets to make the control decision. This new boundary will also require multiple channels for multiple simultaneous conversations if that is occurring in the communication channel. There are significant performance differences between examining single packets as they pass and rebuilding communication streams before analysis. The latter task involves more memory and potentially larger numbers of control states. Combining these two separate tasks in a single device either requires separate devices running in parallel, or a sub-optimal response for single packets.

The law of requisite variety shows how as network complexity grows, the required regulator function becomes unachievable. Problems of enumerating all system states and regulator size do not lend themselves to practical solutions with respect to the scale of complex IT systems in place today. Systems theory points to a solution, that of a smaller system definition, which is done in the network world through enclaves carved out of larger networks via local firewalls. Taking this model to the extreme shows that the point defense of firewalls in front of individual servers is the method to simplify regulator function and provide the highest levels of protection.

Similar problems occur in security regulators designed to detect and control blended attacks. Logging into a system with proper credentials is a normal task, managed and monitored by the access control system. Manipulation of system inputs to achieve a buffer overflow, or SQL injection of other application type attack can in many cases appear as normal traffic to firewalls and access control systems, yet based on user credentials, information transmitted and the context of the commands being issued, can be classified as either authorized or un-authorized. Because of the contextual nature of the attack, this makes the system boundary definitions for the appropriate controller and the required number of monitored states in a controller increase dramatically in many cases. The required number of monitored states in a regulator is an issue that does not scale in a fashion that makes it practical to achieve desired goals with single regulators. In the case of this blended attack, one needs several regulators to control the differing levels of detail. The first control aspect is the case of access control where the controller needs to have a boundary such that it can differentiate between authenticated and non-authenticated users. This boundary may exist in a separate system that is verifying credentials, or it may exist concurrently with access to some specified resource. In distributed systems, having a defined authentication system solves many administrative and scale related issues. The next control boundary would be authorization to access a specific resource such as a web page associated with data retrieval. One of the inputs to the authorization system would be an output from the authentication system, namely some item representing the user. Once the positive authorization is made, the next level of control is applied. Input to a web service, web page or any form for that matter should always be verified before use. This is the best defense against SQL injection type errors. The primary control mechanism against this occurs during the software development process, one of secure code design [13, 14]. Yet this control mechanism exists outside the operational security system.

Add to the equation, the idea that if a form has an input block, that a user is designed to get

there and fill in this information, then previous levels of security only weed out really poorly managed threats. A serious threat will get to the input block – even if only on occasion, for that is the purpose of this aspect of the system: to accept input. Building a layered defense in this situation makes sense, but it requires multiple regulators of differing systems that overlap or converge on this input element. In addition to proper software design processes, specific traps need to be placed at input boxes based on the usage of the material that is input via them. Having input filters that manage input length, filter out scripts and other undesired elements, and that handle values via safe mechanisms, i.e. stored procedures, are a series of layers, yet again, they do not overlap completely. Adding additional protection mechanisms to output mechanisms, to ensure only single records are displayed as opposed to series of records can act as additional control mechanisms. In this case, the control mechanism is measured at one level as granular input and at an entirely different level or regulator as granular output. Understanding how to recognize the need for and implement these different levels of control mechanisms is a result of systems level thinking.

When one examines a complete enterprise, the number of threats and vulnerabilities rapidly soars. Implementing a series of point solutions for each of these different cases is viewed as unsustainable. When faced with a whole bunch of blood thirsty mosquitoes, one has a couple of choices. Kill them one at a time, or in small groups, or move to a different control mechanism (like taking refuge in a car) and exclude them from your reality. Yet, systems theory shows the flaws associated with believing in one size fits all solutions. Endpoint based solutions with highly targeted regulator functions can act very efficiently and with high degrees of specificity against threats. The challenge is to balance numbers of solutions versus unrealistic expectations of single one size fits all solutions.

Systems level theory and thinking opens these opportunities up to designers, moving from the realm of designing numerous similar point solutions to more comprehensive control mechanisms that address the problem from a different angle. People are managed through policies, yet an examination of how this regulator operates demonstrates that it will not be effective against several types of users. Users that are ignorant of the policy will break it not out of malice, but ignorance. Training and awareness will fix some of these, but how effective is this line of defense? Policies may not be enforceable against all parties. Once an employee has decided that they are leaving, whether for another job or not, many of the penalties associated with policies have little strength. Senior management members, those with the greatest span of control over information are also least likely to be directly affected. Either because of position or parachute clause, senior executives are not driven by fear from policies. Contract workers, workers with a grievance, temporary workers – each has its own motivators and controlling agents, yet how many of these are accurately modeled into the control mechanism of the regulator. The rule from requisite variety, that the regulator must have as many control states as the system can have in its operation is applicable here. How many different states can executives, disgruntled employees, temporary workers, etc. exist inside a policy driven system? What are the requisite levels and numbers of controls against the diverse group? Designing the system for the typical employee will work against them, but is this really the threat one desires protection from?

Systems level thinking, backed by an understanding of systems theory is a valuable new way of examining issues and applying the correct balance of solutions. The use of these concepts as tools to assist designers to understand the ramifications of design choices will result in better more secure designs. The concept of regulator scope and environmental boundary relationships are aspects of the security problem that are many times overlooked or brushed aside with sweeping thoughts of it can be fixed with a few more rules. Understanding the true nature of the limitations of a regulator function based on system size brings a notion of reality and theory based foundation into play.

6. Implications

The use of systems level thinking reveals many interesting things about designing security. Examining threats against a system as a collection of individual items in isolation does not provide an appropriate level of information to properly craft a strong defense. In addition to just enumerating the threat and its mechanism, an understanding of the system level that it is invoking will enable the security designer to employ multiple layers of appropriate defenses

that have an ability to impact the threat. Threat modeling is a relatively new technique designed to help security professionals understand and communicate critical issues concerning threats, vulnerabilities and mitigating actions. But this is an analysis of only one side of the equation. Balancing out the threat side is a systems level analysis of the defensive side, where the range of control of regulators and security control boundaries are placed.

Systems employed in a modern business environment today are rarely created from the ground up. Most are enhancements and modifications to previous systems, expansions and changes to even major components as the business shifts and morphs the IT infrastructure to meet the dynamic demands of the business place. This ever changing landscape demands a good understanding of all aspects of the security equation, both from the threat side and the defensive side. With this understanding, it becomes possible to keep security in the picture as a system is upgraded to meet future challenges, both from business needs and security needs.

Developing and maintaining a comprehensive set of systems theory based maps detailing security mechanisms and the effects of boundaries, regulators and the actual elements being regulated will enable security architects to examine entire systems. With this type of information, it can be seen where defense in depth is actually occurring and where it isn't. These diagrams can also assist in the understanding of complex security appliances that are comprised of multiple independent devices on the inside. Both dependent and independent security mechanisms can be mapped and understood. Not only does this increase the simplicity of a previously complex system, it also can expose security system weaknesses that may be obscured by complex designs.

The addition of systems theory understanding adds another tool to the toolbox for designers, developers, architects and security professionals. Systems theory provides a basis for understanding the ramifications of how things work and what the relationships are between a system, its environment and how much can be controlled. This being said, systems theory is not a be all, end all, solution to the myriad of problems facing the security community. It won't solve all problems, nor will it necessarily provide new ideas to use. It will provide a means to understand what can and can't be expected under certain circumstances and provides a means to measure claims of universal coverage with respect to regulator capability.

7. References

[1]M. Abadi, Burrows, M., Kaufman, C., Lampson, B., "Authentication and delegation with smart-cards," *Science of Computer Programming*, vol. 21, no. 2, 1993, pp. 91-113.

[2]A. Adams, Sasse, M. A., "Users Are Not The Enemy," *Communications of the ACM*, vol. 42, no. 12, 1997, pp. 41-46.

[3]W. Ashby, *Introduction to Cybernetics*, Chapman & Hall, 1956.

[4]W. Ashby and R. Conant, "Every Good Regulator of a System must be a Model of that System," *International Journal Systems Science*, vol. 1, no. 2, 1970, pp. 89-97.

[5]M. Burrows, Abadi, M., Needham, R., "A Logic of Authentication," *Proceedings of the Royal Society of London*, no. Series A, 426, 1989, pp. 233-271.

[6]H. Cavusoglu, B. Mishra, and S. Raghunathan, "The Value of Intrusion Detection Systems in Information Technology Security Architecture," *Information Systems Research,* , vol. 16, no. 1, 2005, pp. 28-46.

[7]P. Checkland, "Soft Systems Methodology: A Thirty Year Retrospective," *Systems Research and Behavioral Science*, vol. 17, 2000, pp. S11-S58.

[8]CompTIA Research, *Committing to Security: A CompTIA Analysis of IT Security and the Workforce.*, CompTIA Research, 2002.

[9]W.A. Conklin, G.B. White, C. Cothren, D. Williams, and R.L. Davis, *Principles of*

Computer Security: Security+ and Beyond, Information Assurance & Security Series, ed. Corey Schou, Burr Ridge, IL: McGraw Hill, 2004.

[10]D.E. Denning, "An Intrusion-Detection Model," *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, vol. SE-13, no. 2, 1987, pp. 222-232.

[11]Ernst and Young, Global Information Security Survey 2004. 2004,

[12]L.A. Gordon, M.P. Loeb, W. Lucyshyn, and R. Richardson, *2006 CSI/FBI Computer Crime and Security Survey*, Computer Security Institute, 2006 2006.

[13]M. Howard, D. LeBlanc, and J. Viega, *19 Deadly Sins of Software Security* McGraw-Hill Osborne Media, 2005.

[14]M. Howard and S. Lipner, *The Security Development Lifecycle*, Secure Software Development Series, Redmond, Washington: Microsoft Press, 2006.

[15]D. Ilett, "30,000 botnets march across the Internet," *ZDNet UK*, Barcelona, Spain, November 5, 2004 2004, p. 1.

[16]G.P. Im and R.L. Baskerville, "A Longitudinal Study of Information System Threat Categories: The Enduring Problem of Human Error," *The DATA BASE for Advances in Information Systems*, vol. 36, no. 4, 2005, pp. 68-79.

[17]S. Kamara, S. Fahmy, E.E. Shultz, F. Kerschbaum, and M. Frantzen, "Analysis of Vulnerabilities in Internet Firewalls," *Computers & Security*, vol. 22, no. 3, 2003, pp. 214-232.

[18]J. Luftman, "Key Issues for IT Executives," *MISQ Executive*, vol. 4, no. 2, 2005, pp. 269-285.

[19]C.P. Pfleeger and S.L. Pfleeger, *Security in Computing*, Prentice Hall Professional Technical Reference, 2002.

[20]J.H. Saltzer and M.D. Schroeder, "The Protection of Information in Computer Systems," *Proceedings of the IEEE, 63(9), pages 1278-1308, September 1975*, vol. 63, no. 9, 1975, pp. 1278-1309.

[21]B. Schneier, *Applied cryptography*, Wiley New York, 1996.

[22]Software Engineering Institute, Build Security In. 2006, Strategic Initiatives Branch of the National Cyber Security Division (NCSD) of the Department of Homeland Security (DHS) https://buildsecurityin.us-cert.gov/portal/:

[23]I. Sommerville, *Software Engineering*, 6th ed., International Computer Science Series, ed. A D McGettrick, Essex, England: Pearson Educational Limited, 2001.

[24]M. Theoharidou, S. Kokolakis, M. Karyda, and E. Kiountouzis, "The Insider Threat to Information Systems and the Effectiveness of ISO 17799. ," *Computers & Security*, vol. 24, 2005, pp. 472-484.

[25]Town Hall Meeting, "A Conversation with Greg Garcia, Assistant Secretary for Cyber Security and Telecommunications," in *Proceedings of RSA Conference US 2007*, San Francisco, CA, 2007.

[26]J.J. Whitmore, "A method for designing secure solutions," *IBM Systems Journal*, vol. 40, no. 3, 2001, pp. 747-768.