International Journal of Computing and Business Research (IJCBR)

ISSN (Online) : 2229-6166

Volume 2 Issue 1 2011

ANALYSIS OF NETWORK VULNERABILITIES AND MANAGEMENT TO ENSURE MAXIMUM SECURITY

Dr. S. N. Panda
Professor, RIMT-Regional Institute of Mgt & Tech.,
Mandi Gobindgarh Punjab

Narinder Singh Rana
Research Scholar,
Punjab Technical University,
Jalandhar, Punjab

Shant Kaushik
Asst. Professor,
D.A.V. College
Ambala City, Haryana

Dr. Rajinder Singh
S. D. College
Ambala Cantt., Haryana

ABSTRACT

The numbers of vulnerabilities encountered on the internet are increasing with every passing day. The vulnerabilities of internet come in a wide range of spectrum but they can more of less be categorized in the realms of Hardware, Software and Human-ware vulnerabilities. Though at present the software vulnerabilities lead to maximum security

incidents, but as the organizations are diligently trying to prevent them it is becoming harder for the attacker to exploit them so they are turning there attention to hardware and human-ware vulnerabilities. This paper highlights different domains of vulnerabilities and ways to manage each of them to ensure maximum security of the network and attached resources and minimize the impact of exploit if one occurs.

Introduction

The dictionary meaning of the word vulnerability is "open to attack or damage" and the widely excepted definition of computer security vulnerabilities is "a weakness in a system which allows an attacker to violate the integrity of that system" this definition essentially states that vulnerability is an defect or an error in a computer system which can be exploited by an attacker or intruder to compromise the system. Vulnerabilities exist for various reasons, such as poor security practices and procedures, inadequate training for employees using the system and individuals responsible for network security, and software and hardware products which have not been developed with security as a primary objective.

The computer technology is advancing so rapidly that it is almost impossible to eradicate vulnerabilities altogether; the best one can do, is to simply minimize them. Computer systems are vulnerable due to both technological and human factors. Technically, systems can be affected by software vulnerabilities in various operating system and the application programs and also, because of some hardware vulnerability in the embedded software, or a design flaw in the computing device itself, but according to Kevin Mitnick the weakest link in the network security chain are always the human beings who use the computer system, which is basically due to inadequate training of the employees who use the computer system, the user do not normally follow the best practices while working, for instance they might not use very secure password or might use the password which can easily be broken using brute force attack or they might be a target of social engineering attack because of which they might let out the confidential information to the attackers.

In this paper we have discussed all the three areas of vulnerabilities i.e., Hardware, Software, and Human and show how each might be exploited by the attacker and what are the methods that the organization and individuals should adopt to protect against these vulnerabilities.

Hardware Vulnerabilities

Most computer literate people assume that only complex software contains vulnerabilities and if we can secure all the software vulnerabilities, our computer system will be completely secure but in this assumption we neglect the concept of Hardware vulnerability. In computer literature "Hardware Vulnerabilities" is a collective term used to describe the vulnerabilities in the hardware components and the embedded software used in the computational devices. Hardware vulnerabilities can be explained with help of the hardware exploits that have been used by the hackers over past few years for example in a security conference, security expert Brandan O'Connar discussed a vulnerability that affected Xerox printers [1]. O'Connar exploited this vulnerability in the printer to intercept the data that was sent to the printer for printing and he explained the potential problems created by security deficiencies in embedded software. O'Connor points out that users trust printers, and that a large volume of sensitive information goes through them. In fact a patch for this vulnerability was issued by Xerox but still the problem has not been completely resolved and the printer is still vulnerable to infiltration.

Mostly still in theory, but a large array of hardware errors can lead to security vulnerabilities for instance now one of the inventors of the RSA public key encryption algorithm has conjectured that errors in hardware design of microprocessors can lead to exploitable errors in hardware. A subtle math error would make it possible for an attacker to break the protection afforded to some electronic messages by a popular technique known as public key cryptography. Mr. Shamir wrote that if an intelligence organization discovered a math error in a widely used chip, then security software on a PC with that chip could be trivially broken with a single chosen message.

Executing the attack would require only knowledge of the math flaw and the ability to send a poisoned encrypted message to a protected computer. It would then be possible to compute the value of the secret key used by the targeted system. With this approach, millions of PC's can be attacked simultaneously, without having to manipulate the operating environment of each one of them individually.

Another commonly used and wide spread hardware vulnerability concerns the following fascinating question "Is it possible to maintain secrecy even if an hacker can snoop in your brain?" which precisely means that, can we guarantee security when the most important assumptions of cryptography breaks down [2], i.e. when the hacker can gain access to the internal hardware that is making use of our secrets? The concept of cryptography has made tremendous strides with the increase in computer technology. The strength and complexity of cryptography algorithm is analyzed but the implementations is normally not scrutinized, typically user thinks of a crypto-mechanism as a black box which uses some mathematical function and output the encrypted values and nothing else, however, in practice implementations are not always a true black box partial information about internal computations, either directly or through side-channels, can be leaked leading to a security risk. This subtle difference between practical implementations and theoretical algorithms has led to successful attacks, even when the underlying algorithm was almost perfect. For example, the power consumed during an encryption operation or the time it takes for the operation to complete can leak information about intermediate values during the computation [3, 4], and this has led to practical attacks on smartcards. Electromagnetic radiation [5, 6], compromising emanations [7], crosstalk onto the power line [8, 9], return signals obtained by illuminating electronic equipment [9, 10], magnetic fields [11], cache hit ratios [12, 13], and even sounds given off by rotor machines [14] can similarly give the attacker a window of visibility on internal values calculated during the computation. There are other varieties of attack known as probing attack, where the hacker places some metallic needles in the wire which carry the information and can read the data traveling on the communication media [15]. These side attacks have proven to be a major

security threat to the embedded devices. The lack of any proof-driven cryptography algorithm is due to an inherent assumption called secrecy assumption according to which when ever an message is encrypted it generates only the encrypted value and no information is leaked in this computation, but as we have already seen that there are a bevy of ways in which this secrecy assumption can fail in a real system.

All this being said what is the best security bet against such attacks? What should developers and network security professionals do to prevent these kind of vulnerabilities from being exploited, one possible solution is to use the implementation techniques which ensure that the secrecy assumption will hold true for any given implementation for instance we could consider adding large capacitors to hide the power consumption, switch to dual-rail logic so that power consumption will be independent of the data, shield the device in a tamper-resistant Faraday cage to prevent information leakage through RF emanations, and so on. The firmware in which vulnerabilities are discovered should be patched by upgrading the firmware before the hacker can use it to launch a wide spread attack against the computing devices. While implementing all these counter measures we should remember that all steps are taken to prevent the known and predictable attacks but if the prediction of the possible attacks is incorrect or incomplete the system would still be vulnerable to other kind of side-channel attacks.

Software Vulnerabilities

Software vulnerability is defined as an error in the *specification*, *development*, or *configuration* of operating system, application program or web application whose execution can lead to a security incident. Software vulnerabilities are no doubt the largest and much more exploited vulnerabilities as compared to hardware and human-ware vulnerabilities. The software vulnerabilities include buffer and stack overflow, SQL and code injection, cross site scripting, DoS and DDoS attacks. Some of these vulnerabilities are worth mentioning for instance, according to an estimate of United States Computer Emergency Readiness Team (US-CERT), 20 percent of exploits reported involved buffer overflows. Buffer overflows can damage programs

or compromise data, even when a program is associated with non-administrative account. Developer can inadvertently create buffer overflows in their programs. Buffer overflows vulnerabilities have been found in many open-source programs and in almost all operating systems. Any application or operating system software that has a graphical or command line interface has to temporarily store the user's input. For instance, if a program displays a dialog box requesting a filename, it has to store that filename at least long enough to pass it on to whatever function it calls to open or create a file with that name. In almost all operating system, including Windows, UNIX, and Mac OS X, the computer's random access memory (RAM) is used to store such information. For efficiency the data is put into a stack. In the stack, data is read and removed from memory in the reverse order from which it was put in i.e. last in-first out (LIFO). The data that is put on the stack includes the return address and parameter values used to call a function. Although a program should check the data input by the user to make sure it is appropriate for the purpose intended for example, to make sure that a filename does not include illegal characters and does not exceed the legal length for filenames, frequently the programmer does not bother. The programmer assumes that the user will not do anything unreasonable. Unfortunately, if the user is malicious, he might type in data that is longer than the function parameter allows. Because the function reserves only a limited amount of space on the stack for this data, the result is that the data overwrites other data on the stack. A clever attacker can use this technique to overwrite the return address used by the function, substituting the address of his own code. Then, when the called function completes execution, rather than returning to calling function, it jumps to the attacker's code. Because the user's program executes the attacker's code, the attacker's code inherits the user's permissions. If this user has the super user rights, the attacker can take complete control of the computer, deleting information from the disk, sending emails, and so on.

DoS and DDoS are external attacks as the result of one attack or a number of coordinated attacks, respectively. Designed to slow down or disable networks altogether, these attacks are among the most serious threats that networks face. Security administrators must use tools to monitor network performance in order to catch these threats as soon as possible. Some

monitoring tools are configured to send e-mail or SMS alerts to administrators when such attacks occur, and tools can also be configured to automatically terminate such network access to the attacker. The software vulnerabilities are mainly caused because of the following circumstances:

Flaws in Operating System Design– The designer of operating system use sub optimal policies for user/program management. For instance operating systems does not implement the need-to-know principle and grant some program or user full access to the entire computer system. This operating system flaw allows viruses and malware to execute commands on behalf of the a normal user or super user.

Software Bugs – The programmer leaves an exploitable bug in a software program. The software bug may allow an attacker to misuse an application through various attacking techniques.

Unchecked User Input – The developer assumes that the end user will ony input a valid data. Programs that do not check user input can allow execution of commands or SQL statements from the invalid input which has been maliciously modified to overflow the space resevered for the input field.

In Fig.1 we have shwon the increasing trend of the software vulnerabilities the data from year 1997 to 2008 is the actual data collected from the National Vulnerability Database(NVD) and data from 2009 to 2012 has been pridicted using the polynomial trendline shown in the Fig.2 and the value of R^2 for the trendline is 0.88 and the actual vulnerability graph, Fig 1 & 2 clearly shows that the upward trend in number of vulnerabilities is constantly increasing and may reach a level of almost 14,600 vulnerabilities anually by year 2014.

International Journal of Computing and Business Research (IJCBR)

ISSN (Online) : 2229-6166

Volume 2 Issue 1 2011

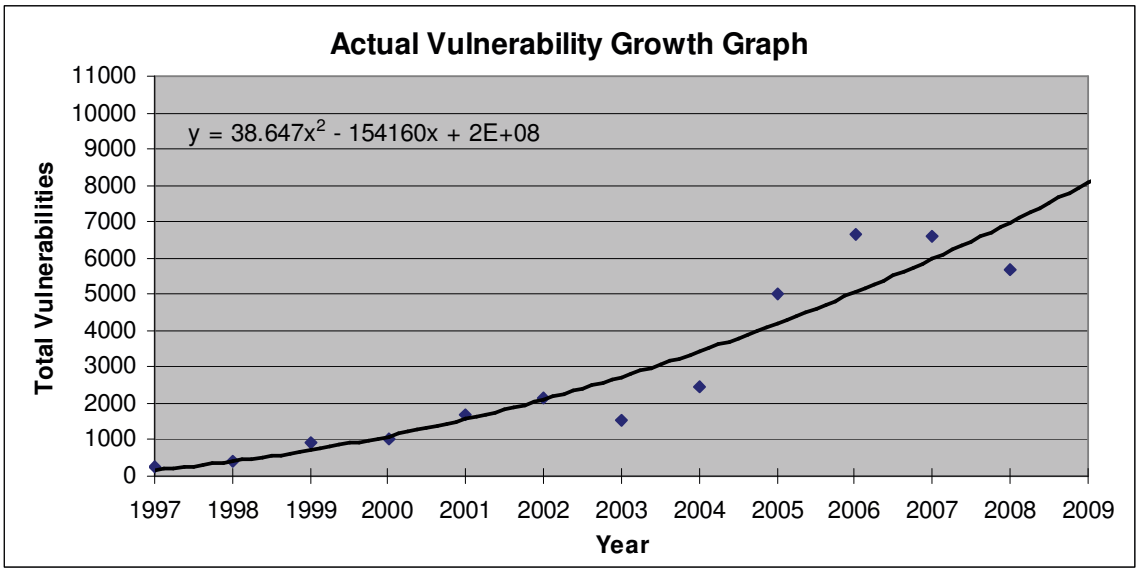

FIG. 1 Software Vulnerability Trend (Actual)

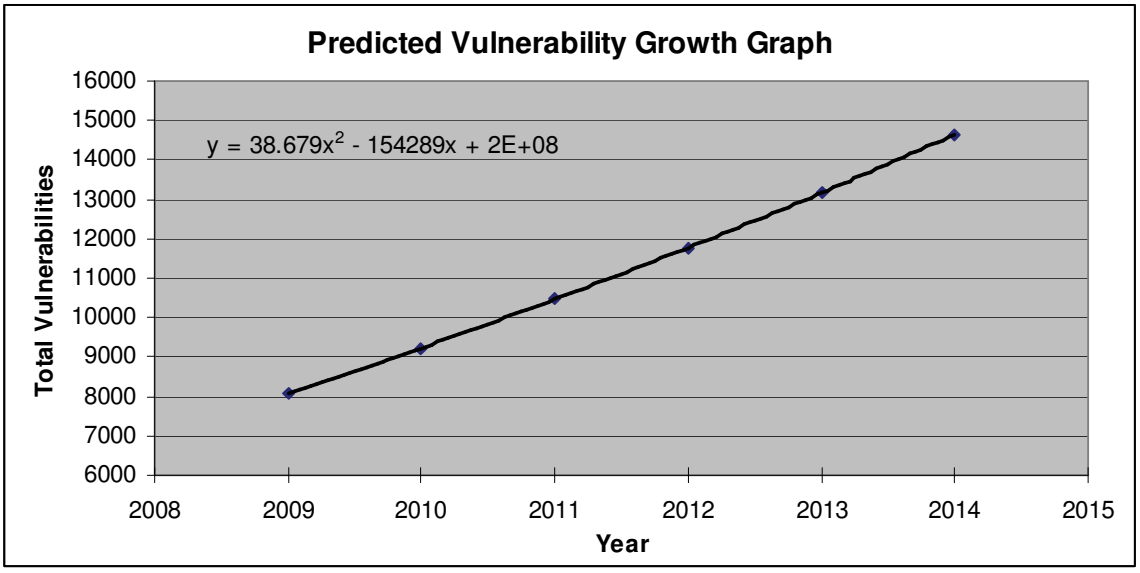

FIG. 2 Software Vulnerability Trend (Predicted)

Software vulnerabilities act as entry gate for various computer attacks. Software vulnerabilities exist for various reasons, for example poor security practices and procedures, inadequate training

for administrators responsible for network security, and software products of poor quality. Sometimes, within some organization or government agencies, an important security patch might not be scheduled for installation on computers until some time after the patches is made available by the vendor. This delay tends to happen if the organization fails to enforce its security policy, if the security department is under resourced, or if the patch to be installed disrupts the computer system when it is installed, which may take inordinate amount of system administrator's time to fix the computer configuration changes made by the installed patch.

Despite these numerous configuration changes caused by the security patches, the effort is worth the return as patching up the operating system as well as all the application and system program is the first and most important defense against software vulnerabilities. As soon as any software vendor would know about the vulnerability in their product ,whether through internal sources or public disclosure, they would create a patch that would plug the vulnerability and distribute it over the internet so, it is extremely important that all such available patches are installed on the system as soon as possible. Also, software vendors are often criticized for commercializing and releasing products with errors. U.S government experts have stated that 80% of successful intrusions into federal computer systems were caused by low-quality software and numerous software errors resulting from too early release. Currently, there is no legal liability or regulatory mechanism relating to the problem of a software producer's selling a product with design defects. The reality is that the licensing agreement accompanying the product includes a disclaimer protecting the software manufacturer from all liability. It is extremely important that such regulation should be implemented so that software companies at least test there products adequately before selling them to the end users. Cooperation between software vendors, standardization bodies and security researchers, as well as sharing information about secure coding or implementing quality certificates for software are some methods that can go a long way in reducing the amount and severity of software vulnerabilities.

Human-ware Vulnerabilities

An organization may have purchased the best security technologies that money can buy, implemented best firewalls, intrusion detection system and other network security products, and always keep there system and software patched but still that company is totally vulnerable. The human factor is the weakest link in the network security. The world's most respected scientist of the twentieth century, Albert Einstein, once said, “Only two things are infinite, the universe and human stupidity, and I'm not sure about the former.” The eminent hackers are usually excellent social engineers who can often get passwords and other sensitive information out of people just by pretending to be someone else. The social engineering attacks can succeed when people are stupid or, more commonly, simply ignorant about good security practices. Many information technology (IT) professionals hold to the misconception that they've made their companies largely immune to attack because they've deployed standard security products - firewalls, intrusion detection systems, or stronger authentication devices such as time-based tokens or biometric smart cards. Anyone who thinks that security products alone offer true security is settling for the illusion of security, they will inevitably, later if not sooner, suffer a security incident. Security is not a technology problem - it's a people and management problem. As developers invent more secure applications, the hacker would find it more difficult to exploit technical vulnerabilities, and it would lead to enhanced exploitation of the human element. Cracking the human firewall is often easy, requires no investment beyond the cost of a phone call, and involves minimal risk.

There is a classic case of deception which is normally used to explain the concept of social engineering. Not many people today still remember the young man named Stanley Mark Rifkin and his little adventure with the now defunct Security Pacific National Bank in Los Angeles. One day in 1978, Rifkin went to Security Pacific’s authorized electronic-transfer room, where the staff sent and received money totaling several billion dollars every day. He was working for a company under contract to develop a backup system for the wire room’s data in case their main computer ever went down. That role gave him access to the transfer procedures, including how bank officials arranged for a transfer to be sent. He had learned that bank officers who were

authorized to order electronic transfers would be given a closely guarded daily code each morning to use when calling the transfer room. In the room the clerks saved themselves the trouble of trying to memorize each day's code they wrote down the code on a slip of paper and posted it where they could see it easily. One day Rifkin had a specific reason for his visit. He wanted to get a glance at that paper. Arriving in the wire room, he took some notes on operating procedures, supposedly to make sure the backup system would mesh properly with the regular systems. Meanwhile, he surreptitiously read the security code from the posted slip of paper, and memorized it. A few minutes later he walked out.

Leaving the room at about 3 o'clock in the afternoon, he headed straight for the pay phone, where he deposited a coin and dialed into the electronic-transfer room. He then changed hats, transforming himself from Stanley Rifkin, bank consultant, into Mike Hansen, a member of the bank's International Department. The conversation went something like this:

"Hi, this is Mike Hansen in International," he said to the young woman who answered the phone. She asked for the office number. That was standard procedure, and he was prepared: "286" he said.

The girl then asked, "Okay, what's the code?"

He responded smoothly, "4789."

Then he went on to give instructions for transferring "Ten million, two-hundred thousand dollars exactly" to the Irving Trust Company in New York, for credit of the Wozchod Handels Bank of Zurich, Switzerland, where he had already established an account.

The girl then said, "Okay, I got that. And now I need the interoffice settlement number."

This was a question he hadn't anticipated, something that had slipped through the cracks in his research. But he managed to stay in character, acted as if everything was fine, and on the spot answered without missing a beat, "Let me check; I'll call you right back." He changed hats once again to call another department at the bank, this time claiming to be an employee in the electronic-transfer room. He obtained the settlement number and called the girl back. She took the number and said, "Thanks."

A few days later Rifkin flew to Switzerland, picked up his cash, and handed over $8 million to a Russian agency for a pile of diamonds. He flew back, passing through U.S. Customs with the stones hidden in a money belt. He had pulled off the biggest bank heist in history--and done it without using a gun, even without a computer. Oddly, his caper eventually made it into the pages of the Guinness Book of World Records in the category of "biggest computer fraud." Stanley Rifkin had used the art of deception--the skills and techniques that are today called social engineering. Incidents like this are happening every day and it is a growing concern for the organization as this kind of fraud or attack might be happening or is imminent. According to a survey by a computer security agency 85 percent of responding organizations had detected computer security breaches in the preceding twelve months, and 64 percent organization reported a financial loss due the security breaches.

We as a human being tend to trust and help each other and the social engineers abuse this trust to get sensitive information out of an ignorant person. Despite our intellect, we humans - you, I, and everyone else - remain the most severe threat to each other's security. Deploying more technology is not going to solve the human security problem. Suppose a social engineer want to get information about the top secret project that your organization is working on. What's going to stop him? Your firewall? No. Strong authentication devices? No. Intrusion detection systems? No. Encryption? No. Limited access to phone numbers for dial-up modems? No. Code names for servers that make it difficult for an outsider to determine which server might contain the product plans? No. The truth is that there is no technology in the world that can prevent a social engineering attack.

So what is the best way to guard against such attacks? The simple answer to this question is security through a consolidate defense of technology, training, awareness, and procedures. According to the companies that conduct security testing state that the attempt to break an organization's security using social engineering attacks is almost 100 percent successful. The best way to mitigate these threats is through the implementation of proper security procedures

that guide the behavior of the employees such as changing the password every fortnight, using strong password, not giving out any company information over the phone without properly establishing the callers identity, training and educating the employees about the tricks used by social engineers to deceive them. The only way to keep the intruders away is to have a trained, aware, and conscientious workforce. It should be emphasized that the constant awareness of the employees is equally important nowadays it is not uncommon for the companies to spent 40 percent of there security budget is such training and awareness programs. The first step is to make everyone in the enterprise aware that unscrupulous people exist who will use deception to psychologically manipulate them. Employees must be educated about what information needs to be protected, and how to protect it. Once people have a better understanding of how they can be manipulated, they are in a far better position to recognize that an attack is underway.

Testing Security-Network Penetration Testing

The software development life cycle can not be completed without the testing phase, which can at times account for anywhere between 30 to 50 percent of total development cost of application development, so there is no good reason that this critical step should be ignored in case of network security deployment. The black box testing of network security can be done using the procedure called network penetration testing in which various automated tools and manual procedures are used to scan the network to identify vulnerabilities, improve the organization security policy and ensure that the security implementation are functioning as required and expected. The penetration test should be executed periodically to uncover the network security weaknesses and the vulnerabilities that can be used by the intruders to compromise the system and launch various attacks such denial of service attacks, Trojans, and other Malware attacks against the organization and testing also expose the additional vulnerabilities that might be introduced by patching the system and a accordingly a corrective measure can be initiated.

Conclusion

The network vulnerability spectrum is wide and the exploits might exist in Hardware, Software and the people using them. The only way to mitigate the chance of a vulnerability being used against a system is through constant vigilance, including careful system maintenance such as applying software and operating system patches, best practices in security deployment such as using firewalls and access controls, auditing both during development and throughout the deployment lifecycle, using the best and secure hardware and firmware, and constant training and awareness of the employees. Finally though most of the companies invest huge amount of fortune in security product and services they do not ensure that the procedures and policies are properly implemented. So the organizations need to ensure that the amazing polices that have been formulated are indeed being followed and the methods such as penetration testing should be used detect any diversion from this and prevent the loss or compromising of sensitive information.

References:

1. Brendan O'Connor “Hacker warns of hardware vulnerabilities,” Black Hat security conference, Aug 2006.
2. Yuval Ishai, Amit Sahai, and David Wagner, “Private Circuits: Securing Hardware against Probing Attacks”.
3. P. Kocher, “Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems,” CRYPTO’96, Springer-Verlag, 1996, pp.104–113.
4. P. Kocher, J. Jaffe, B. Jun, “Differential Power Analysis,” CRYPTO’99, Springer-Verlag,1999, pp.388–397.
5. J.-J.Quisquater, D.Samyde, “ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards,” Esmart 2001, LNCS 2140, Springer-Verlag, 2001.
6. K.Gandolfi, C.Mourtel, F.Olivier, “Electromagnetic Analysis: Concrete Results,” CHES’01, LNCS 2162, Springer-Verlag, 2001.

7. W.VanEck, "Electromagnetic Radiation from Video Display Units: An Eavesdropping Risk," Computers & Security, v.4, 1985, pp.269–286
8. D. Wright, Spycatcher, Viking Penguin Inc., 1987.
9. US Air Force, Air Force Systems Security Memorandum 7011—Emission Security Countermeasures Review, May 1, 1998.
10. R.Anderson, M.Kuhn, "Soft Tempest: Hidden Data Transmission Using Electromagnetic Emanations," Proc. 2nd Workshop on Information Hiding, Springer, 1998.
11. J.-J.Quisquater, D.Samyde, "Eddy current for Magnetic Analysis with Active Sensor," Esmart 2002, Sept. 2002.
12. J.Kelsey, B.Schneier, D.Wagner, "Side Channel Cryptanalysis of Product Ciphers," ESORICS.
13. D.Page, "Theoretical Use of Cache Memory as a Cryptanalytic Side-Channel," Tech. report CSTR-02-003, Computer Science Dept., Univ. of Bristol, June 2002.
14. D.Kahn, The Codebreakers, The MacMillan Company, 1967.
15. R.Anderson, M.Kuhn, "Tamper Resistance—A Cautionary Note," USENIX E-Commerce Workshop, USENIX Press, 1996, pp.1–11.
16. J.R.Rao, P.Rohatgi, "EMpowering Side-Channel Attacks," IACR ePrint 2001/037.
17. S.Chari, C.S.Jutla, J.R. Rao, P. Rohatgi, "Towards Sound Approaches to Counteract Power-Analysis Attacks," CRYPTO'99, Springer-Verlag, 1999, pp.398–412.
18. US-CERT, Monthly Activity Summary, September 2008.
19. Hang Chau, Network Security – Defense Against DoS/DDoS Attacks.
20. Kevin Mitnick - The Art of Deception.
21. Dr Peter Shabad, "To Expose or to Cover Up: Human Vulnerability in the Shadow of Death," (2006), Contemporary Psychoanalysis, 42:413-436.
22. Mr. Dennis Hughes, Quoting presentation to computer security conference by FBI official, February 3, 1997.
23. National Security Institute, "Top Hacker Tells Congress that Employees Are Security's Weakest Link," NSI Advisory, April 2000.

24. Ashok Kumar, S.N.Panda, "CPIH-An effective incidence Handler at Convergence Point" PCTE Journal of Computer Sciences", Vol3 Issue no. 2. Dec-2007.