Toward A Secure System Engineering Methodology*

Chris Salter, NSA, salter@thematrix.ncsc.mil
O. Sami Saydjari, DARPA, ssaydjari@darpa.mil
Bruce Schneier, Counterpane Systems, schneier@counterpane.com
Jim Wallner, NSA, jmwalln@missi.ncsc.mil

Abstract

This paper presents a methodology for enumerating the vulnerabilities of a system, and determining what countermeasures can best close those vulnerabilities. We first describe how to characterize possible adversaries in terms of their resources, access, and risk tolerance, then we show how to map vulnerabilities to the system throughout its life cycle, and finally we demonstrate how to correlate the attacker's characteristics with the characteristics of the vulnerability to see if an actual threat exists. Countermeasures need to be considered only for the attacks that meet the adversaries' resources and objectives. Viable countermeasures must meet user needs for cost, ease of use, compatibility, performance, and availability.

---

* This paper is based on research done by a working group sponsored by the National Security Agency.

Toward A Secure System Engineering Methodology

1.0 Introduction

Twenty years ago, corporate and military information infrastructures were separate and distinct. Today they are one and the same. The military depends upon the same computer networks and networking equipment to fight wars as industry depends upon to conduct business. The government has two roles with respect to its nation's information infrastructure: to be forthcoming about the genuine threat, and to foster public confidence. By drawing on the National Security Agency's experience in attacking and defending information systems, this paper opens a dialogue among academia, industry, and government toward securing the global information infrastructure.

The security community needs a common vocabulary to discuss threats and countermeasures [DSB96], and a common methodology to discover weaknesses in systems, to prioritize weaknesses in terms of relative dangers to the system, and to determine cost-effective countermeasures [Amo94]. The purpose of this methodology is not to "penetrate and patch" a system, but rather to discover the sources of system weaknesses and to uncover reasonable design strategies to create stronger systems. This paper builds upon previous models, such as dynamic threat analysis [Smi89] and security vulnerability analysis [Wei91].

Before being able to design secure systems, designers must thoroughly understand the means, motives, and opportunities of adversaries. First, the system engineer derives the list of possible adversaries determined by who stands to profit in attacking the system. Next, the engineer characterizes adversaries in terms of their available resources, access to the targeted system, and risk tolerance. Then, the system engineer maps out how information flows throughout the system. Studying the information flow allows the engineer to discover all the critical components and procedures. Components are vulnerable during design, production, distribution, use, and retirement. Each step of a procedure must be scrutinized. The system engineer determines when and how an adversary can gain access at each point in time. For each access point, the engineer considers what an adversary could accomplish, within the bounds of his resources and objectives. Countermeasures are only needed for attacks that meet the adversaries' resources and objectives. To be useful, countermeasures must meet user needs for cost, ease of use, compatibility, performance, and availability.

The paper has three major parts. The first part models adversaries, the second part models vulnerabilities, and the third part provides a secure system engineering methodology. The final section applies the methodology to a secure telephone as an example.

2.0 Adversary Model

Before even looking at the security properties of a system, it is necessary to build a model of adversaries and their characteristics. Our Adversary Model characterizes adversaries in terms of their resources, access, risk tolerance, and objectives. Resources and access determine what adversaries can do, risk tolerance determines what they are willing to do, and objectives determine what they want to do. Different types of adversaries can be classified this way: hackers, malicious insiders, organized crime, industrial competitors, terrorists, and national intelligence organizations. For example, a hacker has low resources, low access, and moderate risk tolerance when compared to an infowarrior who has significant resources, high access, and high risk tolerance [Sch98].

A rational adversary follows the path of least resistance; that is, he chooses an attack that maximizes his expected return on investment, given his budget constraints of resources (money, expertise, manpower, and time), access, and risk. Some attacks require a great deal of access and risk, but not much expertise: e.g., a car bomb. Other attacks require a great deal of computational power, but minimal access and less risk: e.g., a brute-force search of the key space of an exportable encryption algorithm. Financial resources are not directly considered in this model, since they are used to buy either capabilities or access: a rich adversary can trade money for access by paying off an insider, for expertise by buying technology, and for risk by executing a more sophisticated, less intrusive attack.

Different adversaries value attack goals differently. For example, a criminal adversary might value the goal of defrauding a system above simple denial of service attacks, while a malicious insider might be more interested in denial of service. Some of the recent attacks against web browsers were motivated more by publicity than anything else [Ale97]. Attacks that might not be profitable or even rational to a criminal might be perfectly reasonable to someone interested only in exposing weaknesses in a particular product.

The defender might value an asset completely differently than the attacker. An illustrative example from the bricks-and-mortar world is supermarket shopping carts. These carts cost almost $100 to the supermarket and are worth defending. When cart theft becomes a problem, supermarkets require a 25 cent deposit; the carts simply are not worth 25 cents to the attacker.

3.0 Vulnerability Model

After characterizing attackers as described above, the next step is to map the vulnerabilities of a system. The metaphor we use to describe the totality of vulnerabilities is called the Vulnerability Landscape. A vulnerability landscape is made up of peaks and valleys: the valleys represent vulnerabilities and the peaks represent countermeasures. The deeper the valley, the greater the vulnerability. The higher the peak, the more effective the countermeasure. The defender must build a wall to protect the entire system; he cannot build a single large defensive tower and hope that the attacker runs into it.

To understand the vulnerability landscape, the security engineer first maps out the target system. This includes the system itself, as well as things related to it: physical facilities, personnel and their responsibilities, and equipment and procedures used to handle information.

This landscape is built up over time, as more data accumulates and the system matures. The system engineer does not have to recreate the entire landscape whenever changes are made to the system or new weaknesses are found. Instead, the engineer can make changes to the model, allowing the model to be built up over time.

Any successful attack has three steps: One, diagnose the system to identify some attack. Two, gain the necessary access. And three, execute the attack. To protect a system, only one of these three steps needs to be blocked. The goal for the defender is to find the most cost-effective way to block any potential attack. He must estimate the adversary's risk tolerance and willingness to expend resources to gain access and execute an attack. Of course, blocking multiple steps, or blocking a single step multiple ways, provides defense in depth and more assurance against a flaw in a countermeasure. However, not all system engineers have the luxury of implementing that many countermeasures.

The vulnerability landscape sorts the weak points by access. To discover all the opportunities for access, hence all the opportunities to attack, the system engineer must consider the entire **life cycle** of each component of the security environment. The two access categories are **physical security** and the **trust model**. Then the system engineer can construct, from the access points, how the adversary executes an attack to meet the objectives. Finally, the system engineer finds countermeasures that block at least one of the three attack steps.

3.1 The Life Cycle

Assume an adversary decides to bug the telephones or copiers destined for a company's offices. The adversary has many opportunities to conduct an attack. The office equipment is vulnerable during its entire life cycle: on the drawing board, in the manufacturing plant, on the loading dock, in the work place, or (in memory devices [Gut96], for example) even after disposal. Depending on the available access, the adversary may alter or swap the equipment during production, shipment, installation, normal operations, or maintenance. For example, the Soviets placed a transmitter in a plaque presented as a gift to the U.S. ambassador in Moscow.

The work environment of the virtual world is software running on networked computers, and attacks against software systems can be just as varied. A software developer might inadvertently leave an exploitable bug in the latest release of its system. An adversary could put a Trojan horse in a popular net browser and distribute it for free over the Internet. An adversary could write a virus that attacks accounting software and delivers it in an executable attachment to an e-mail message. System engineers must secure the entire software life cycle.

All components of an information system have a life cycle. For example, a vendor programs an e-mail client and ships it to a distributor. The distributor copies and forwards the product to a wholesaler or retailer. The retailer eventually delivers a product into the user's workplace. Note the recursive nature of the life cycle. By answering questions such as who was the designer, where was it made, who delivered it, who has access once it is deployed, and finally, what happens to it when discarded, the system engineer can burrow down to points of vulnerability. The table below provides examples of attacks during a system's life cycle.

Life Cycle Phase	**Definition**	**Example adversary behavior**
Design	From initial idea to design specs	subtly alter system specification to create a flaw
Production	From build-to specs to roll-out	substitute security-critical chip on production line
Deployment	From roll-out to transit to delivery to user	substitute system unit while in transit with bogus unit
Operation and maintenance	From delivery to maintenance to retirement	insert malicious code into application, OS, or network
Destruction	From retirement to destruction	extract stored key from unit to read back-traffic

3.2 Physical Security

Physical security speaks to the problem the world has been trying to solve since the beginning of civilization: how to enforce the notion of ownership. Fences, locks, and guards are all tools of physical security. Today's organizations have considerable expertise implementing physical security measures commensurate with the physical threat; they know their adversaries and what countermeasures are sufficient to protect their assets.

In the physical world, layered security solutions reinforce each other. Behind the fence, guards patrol the perimeter of a locked building. A $5 lock on the door may be sufficient given the guard and the fence. A $10 lock may be wasteful given the open window nearby.

In the physical world, an adversary who wishes to masquerade as a trusted member of a community takes the personal risk of being found out and apprehended. In the virtual world, a spy can cross the border and impersonate a trusted member of the organization with less risk of being detected or physically apprehended. From anywhere in the world, the spy can eavesdrop on electronic conversations or break into private networks and steal information without fear of apprehension. Installing a firewall is analogous to building walls and locking doors. Encryption creates a private room in cyberspace for a confidential conversation or an electronic safe for stored information.

3.3 The Trust Model

The trust model represents how an organization determines whom to trust with its assets. For example, before becoming trusted employees, applicants might have their résumés verified, their references interviewed, and their criminal records checked. Once employed, picture identification badges and parking stickers might be issued. In the physical world, it is easy to identify those individuals who are trusted and those who are not.

The problem is how to extend the trust in individuals from the real world to the virtual world. Traditional biometrics, such as voice and face recognition or handwriting, do not work without the physical presence of the individual to draw upon. Presently, automatic systems only rely on passwords and digital certificates.

Trust relationships describe how different parties in the system trust each other. Some systems require a trusted party to operate the system (the Kerberos login protocol [KN93], for example), while others only require trusted parties for the initial design, manufacture, and installation of the system. Some systems require a great deal of trust—the trusted party holds a copy of the encryption keys—while other systems only require a trusted party for authentication, or validation, or a timestamp. All systems have many levels of trust in many different areas. Any one of these trust relationships is a potential vulnerability.

3.4 A Rational Response to the Vulnerability Landscape

Protective countermeasures should be applied evenly across the landscape to prevent the attacks that pose the greatest threat. Blocking just one of the steps necessary for a successful attack sufficiently deters an adversary. A system engineer should choose the most cost-effective countermeasures: i.e., it doesn't make sense to spend more money improving the locks on the front door when the adversary is apt to break through the glass window. It also doesn't make sense to spend $100 on protective countermeasures to protect $10 worth of assets. Simple countermeasures, including education, policy, and procedures, are rational, cost-effective means of mitigating the risks posed by a vulnerability. These non-technical countermeasures can significantly raise the risk and sophistication needed by the adversary to execute a successful attack.

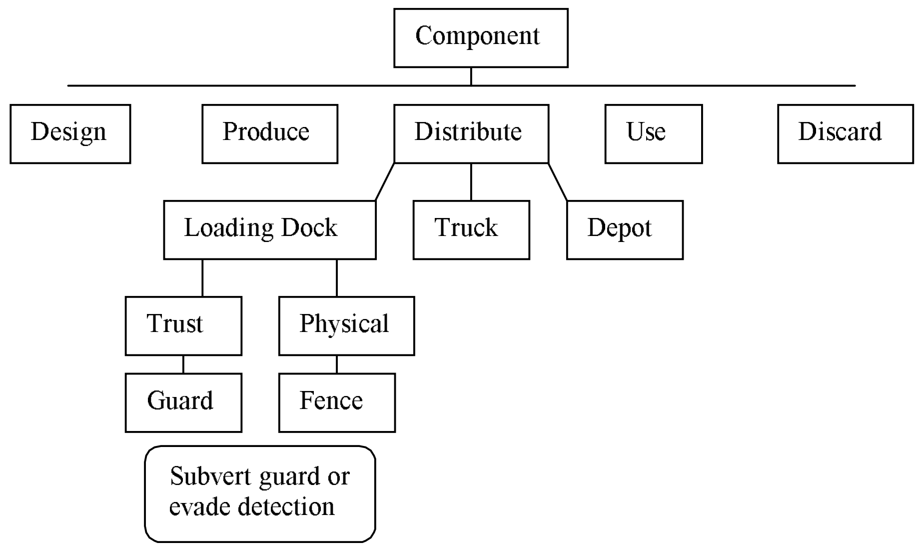
4.0 Methodology Overview

This section provides a methodology for characterizing attacks and choosing rational countermeasures. This methodology is based on an "attack tree" model. An attack tree is a visualization tool to enumerate and weigh different attacks against a system.

Our methodology has five broad steps.

1. Create attack trees for the system.
2. Apply weights to the leaves.
3. Prune the tree so that only exploitable leaves remain.
4. Generate corresponding countermeasures.
5. Optimize countermeasure options.

Step 1) The system engineer systematically creates an attack tree by replicating the work of an adversary to find the weak points in a system. The root node of our tree is the component that prompted the analysis. (Note that the root node could be different, e.g., the root node could be the objective of the adversary.) To form the child nodes, the system engineer decomposes the node into its life cycle. Each phase in the life cycle breaks down into the two access categories, physical security and trust model. If appropriate, each node is again decomposed in this manner. The analysis terminates in a series of vulnerability leaves.



Step 2) For each leaf in the attack tree, the system engineer assigns qualitative values for risk, access, and cost to the adversary. For example, a possible node for encrypted messages would be passive collection and brute force decryption of the ciphertext. The risk of the attack is low since it is done safely at a distance, the access required is low, and the cost depends upon the strength of the cryptographic algorithm.

Step 3) The system engineer then prunes the trees. Countermeasures are needed only for those attacks that meet an adversary's objectives, matches his capabilities, and offer a sufficient return. For example, if a particular attack requires 2^{256} bytes of computer memory, it could safely be pruned as being beyond the resources of any adversary. If another attack requires access to a closed communications system, such as a stand-alone network, it might be beyond the reach of some adversaries but within the capability of others.

Step 4) The system engineer determines countermeasures for the most exploitable nodes.

Step 5) The system engineer ranks the countermeasures with respect to its five attributes:

(1) cost to purchase and run,
(2) ease of use,
(3) compatibility with in-place technology and ability to interoperate with other communities of interest,
(4) its overhead on system resources and, finally,
(5) time to market or availability.

Then, he attempts to deploy the most cost-effective set of countermeasures to defend against the vulnerabilities identified in step (3). Achieving optimal coverage is hard. Given the coarse granularity of the weights, it is futile to develop the algorithm that transforms numbers into the optimal countermeasure set. Still, the model allows the analyst to make an informed choice. A good model depends upon smarter analysts, not smarter software. A software tool that implements this methodology can only enhance the creativity of the system engineer in finding vulnerabilities and finding countermeasures.

4.1 Example: Sensitive Phone Calls

To better explain our methodology, we include this example. We make several simplifying assumptions, and don't analyze some aspects of the system. The point of this example is to demonstrate how to think about the problem, not to be complete.

A large organization (either a national defense organization, or a multinational corporation) needs to make sensitive phone calls. Its adversaries would like to have the information in these calls. In this case, the foremost adversary is a foreign intelligence agency [Fia97,Win97]. This adversary takes moderate risks and has the resources and access to carry out highly sophisticated attacks.

The attack tree for a public phone system can be quickly trimmed to one attack node. To exploit the sensitive phone call, the adversary has to intercept and collect the telephone signal, either by tapping wires, intercepting microwave links, or collecting cellular traffic. In the modern telephone network, the risk associated with this passive intercept is low, since the collection can often be done at a distance and leaves no footprint. However, producing timely intelligence reports from bulk intercepts is expensive. Still, intelligence that comes directly from the source is usually highly accurate and very valuable. Thus national intelligence will make the large investment in these attacks, requiring defenders to find countermeasures.

	Risk	Access	Cost	Effectiveness
Bulk Intercept	L	L	M	H

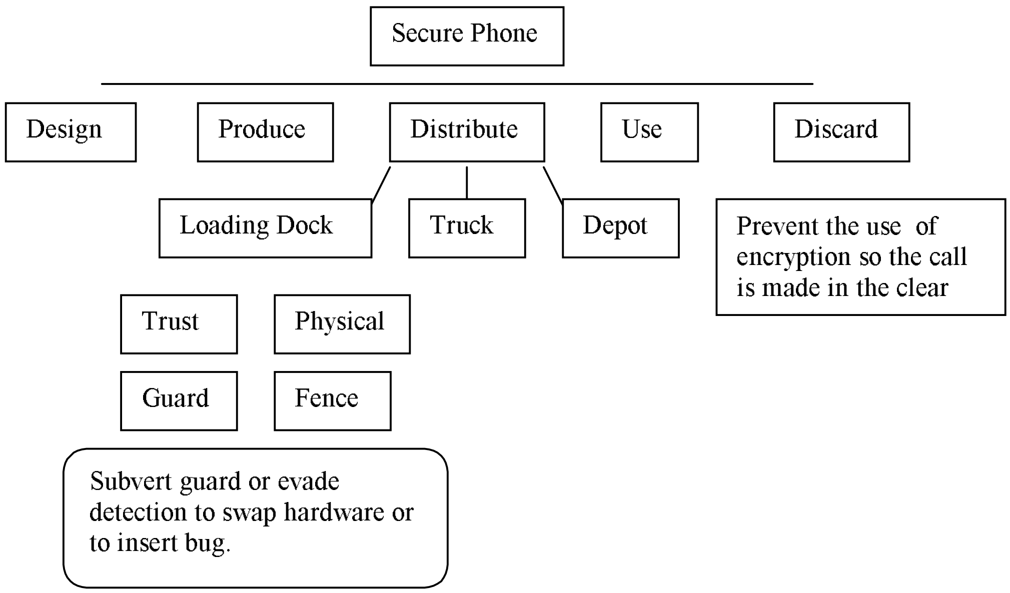
To respond to the threat of bulk intercept, the system engineer has three alternatives:

1. Do nothing. Allow users to make sensitive calls on the public phones.
2. Place encryption in the telephones used for sensitive phone calls.
3. Use dedicated encrypted transmission lines between physically secure enclaves.

	Cost to Implement	Effectiveness
Do Nothing	**Negligible**	**Insignificant**
Encryption in phone	**Considerable**	**High**
Dedicated lines	**Exorbitant**	**Limited**

The price (to the organization's security) of allowing the adversary to intercept unencrypted, sensitive calls is too high, and the price (to the organization's budget) of installing dedicated communications lines for the entire organization is exorbitant. Therefore, the secure phone with encryption is the most cost-effective solution.

The introduction of a secure phone into the communications system creates new vulnerabilities that must now be evaluated. The system engineer follows the five steps of the methodology above to evaluate the new vulnerabilities in the system.



Step 1) **Construct the attack tree.** One possible attack is to modify the secure telephones. To gain access to a shipment of phones on the loading dock, the adversary can jump a fence or bribe a guard. He can then modify the phone so that when the sensitive call is made, he can listen to the conversation. For example, he can alter the randomizer so that the encryption key will be guessable so he can then decrypt the ciphertexts [KSWH98]. Alternatively, he could install a transmitter in the handset to broadcast the unencrypted conversation.

Variations of these two attacks can be executed in all phases of the secure phone's life cycle. For example, during the design phase, an insider could subvert how the phone generates keys [KSWH98] or could introduce a subliminal channel that leaks the key.

Another attack that is effective against secure telephones is to deny the use of the encryption algorithm, e.g., by jamming the public key exchange during the preamble. If the secure mode does not work, the typical user will fall back to the insecure mode [Nee93], and have a sensitive phone call in the clear.

Another attack that is not effective is a brute-force search of the key space (unless the size of the key has been artificially restricted).

Step 2) **Apply weights to the leaves.** The following table summarizes the relative risk, access, cost, and effectiveness of these attacks.

	Life Cycle	Risk	Access	Cost	Effectiveness
Modify Key Generation	Design	L	H	H	H
	Produce	L	H	H	H
	Distribute	L	M	H	L
	Use	L	H	H	M
Subliminal Channel	Design	M	H	M	H
	Produce	M	H	M	H

	Distribute	M	M	M	L
	Use	M	H	M	M
Prevent Encryption	Use	L	M	H	L

Note that attacking a phone with encryption drives up the cost and risk of an attacker when compared to attacking phones without encryption.

Step 3) **Prune the tree so that only exploitable leaves remain.** The system engineer identifies the affordable attacks with respect to cost, access, and risk. As the chart indicates, all the residual vulnerabilities can be exploited by an intelligence agency. Further countermeasures are necessary.

Step 4) **Generate corresponding countermeasures.** The system engineer determines countermeasures. Domestic design and production of equipment increases an adversary's risk of being caught. Building to industry standards for encryption and randomization makes the product more analyzable during the entire life cycle [YY96]. Tamper-evident hardware and packaging increases the adversary's risk of detection, as does digitally signing and verifying any software that can be altered. Random installed units should be broken down and analyzed. Blind buys make it harder for the adversary to identify an opportunity. Sweep teams can detect extraneous signals, and filters can block subliminal channels. Encrypting all phone calls at the switches (not just the sensitive ones) adds another layer of protection; it obscures valuable traffic, driving up the cost of collection and processing. The countermeasure list goes on and on.

Step 5) **Optimize countermeasure options.** The system engineer assesses these countermeasures for their cost, ease of use, interoperability, compatibility, performance, overhead, time to market, and availability. The following is a preliminary list of countermeasures.

Exhibit caution in choosing the manufacturers. By buying security products from a company, you are trusting their design ability, their manufacturing integrity, and their internal security practices. Be especially wary of those companies that the adversary can influence.

Use good standards for encryption and randomization. This might be the most cost-effective approach, since it is easier to analyze standard components. Standards allow the widest design and evaluation; they also increase confidence and drive down the cost of innovation.

Employ sweep teams to detect extraneous signals from the phone. (These teams detect many extraneous signals, and are effective against a wider range of attacks.) Provide filters that narrow the signal for subliminal channels. These two countermeasures can be quite cost-effective in combination with the proper physical security.

The adversary will attack the weakest point in the information's life cycle. Therefore, the system engineer must analyze the entire life cycle of the information, not just the life cycle of a secure telephone. To complete our analysis of a secure phone, we would construct vulnerability trees for insider threats, physical space, and key management: i.e., for who handles the information, where the information resides and conversations take place, and how the telephone will be keyed.

4.2 Another Brief Example: Secure E-mail

Encrypting e-mail is significantly different from encrypting phone calls. For phones, the information is at risk only for the duration of the call. For e-mail, which may be stored for considerable lengths of time, the information is at risk also while at rest. Moreover, the adversary can subvert the operating system of the underlying computers to attack the information. The adversary can introduce the attack at a distance, with little physical risk, and can possibly obtain all of the target's information, not just a single message. Finally, the attack can be automated to rapidly exploit a wide range of targets. Any "key escrow" or "corporate message recovery" creates yet another lucrative opportunity for an adversary [AAB+97].

What's worse, the availability of computer networks has become critical for the world to do its work. Back doors into computer networks used by intelligence agents to read mail can also be used by terrorists to destroy the network. Unfortunately, stopping a terrorist from destroying the network is far more difficult than stopping an eavesdropper from reading electronic mail.

5.0 Conclusions

The modern computer network emerged from a partnership between the corporate world and the military, and the resulting technology provides significant advantages to society, both economic and military. To ensure that these networks will be secure enough for business, personal, and military purposes, industry, academia, and government must work together.

Our model tries to capture the commonality between the corporate and military worlds. By characterizing well the actual adversaries and all possible attacks, the system engineer can build systems that are secure against the real-world threats without over-engineering against one particular threat. By understanding what an adversary is most likely to do, the system engineer can invest effectively in countermeasures.

Four distinct communities can benefit from this model. The system engineer can construct the most cost-effective set of countermeasures. The evaluator can systematically find residual vulnerabilities and rationally assess their residual risk. The vendor can invest in improvements that genuinely benefit customers, and thus will help sell products to customers who can make informed buying decisions that minimize their risk and maximize their investment.

Of course, a theoretical model can only go so far. The problems of network security not only require the combined expertise of industry, academia, and government to solve, but also require hands-on experience with actual systems. We should learn from the hacker community and use the power of the information technology to propagate expertise. Those who design, build, evaluate, and use security products must share their experiences with actual threats and vulnerabilities along with their successful countermeasures.

6.0 Acknowledgments

The authors would like to acknowledge the members of the working group that contributed to this paper: Terry Oxford, Jay Turner, Richard George, Don Simard, Neal Ziring, and others from NSA.

Bibliography:

[AAB+97] H. Abelson, R. Anderson, S.M. Bellovin, J.Benaloh, M. Blaze, W. Diffie, J. Gilmore, P.G. Neumann, R.L. Rivest, J.I. Schiller, and B. Schneier, "The Risks of Key Recovery, Key Escrow, and Trusted Third-Party Encryption," *World Wide Web Journal*, v. 2, n. 3, 1997, pp. 241-257.

[Ale97] M. Alexander, *Net Security: Your Digital Doberman,* Ventana Press, 1997.

[Amo94] E. Amoroso, *Fundamentals of Computer Security Technology,* Prentice-Hall, 1994.

[DSB96] Defense Science Board, Report of the Defense Science Board Task Force on Information Warfare-Defense, November 1996.

[Fia97] H.F. Fialka, *War by Other Means,* W.W. Norton and Co., 1997.

[Gut96] P. Gutmann, "Secure Deletion of Data from Magnetic and Solid-State Memory," *The Sixth USENIX Security Symposium Proceedings,* USENIX Association, 1996, pp. 77-90.

[KSWH98] J. Kelsey, B. Schneier, D. Wagner, and C. Hall, "Cryptanalytic Attacks on Pseudorandom Number Generators," in preparation.

[KN93] J.T. Kohl and B.C Neuman, "The Kerberos Network Authentication Service," RFC 1510, Sep 1993.

[LUP+96] L Lund, W. Unkenholz, D. Parks, C. Bowers, H. Brill, B. Garner, “O’er the RAMparts We Watch: An Introduction to the Risk Analysis Model (RAM)”, *National Security Agency,* 1996.

[Nee93] R. Needham, "Denial of Service*," 1st ACM Conference on Computer and Communications Security*, ACM Press, 1993, pp. 151-153.

[Sch98] B. Schneier, *Secrets and Lies: The Myth of Security in the Digital World,* John Wiley & Sons, 1998. (In preparation.)

[Smi89] S. Smith, "LAVA’s Dynamic Threat Analysis," *Proceedings of the 12th National Computer Security Conference,* 1989.

[Wei91] J.D. Weiss, "A System Security Engineering Process," *Proceedings of the 14th National Computer Security Conference*, 1991.

[Win97] I. Winkler, *Corporate Espionage,* Prima Publishing, 1997.

[YY96] A. Young and M. Yung, "The Dark Side of 'Black-Box' Cryptography or: Should We Trust Capstone?" *Advances in Cryptology—Crypto '96 Proceedings,* Springer-Verlag, 1996.