Designing for Disaster: Building Survivable Information Systems

Ronda R. Henning
Harris Corporation

Disaster recovery is a topic traditionally relegated to the operations staff as a mandatory function. Visions of natural disasters and terrorist acts keep information technology managers awake at night planning how to maintain normal business computing resources. But disasters can be as minimal as a broken water pipe, or an end-user who inadvertently unleashes an Internet worm within the enterprise. This article presents an alternative approach: designing survivability measures into an information system from the start. A discussion of survivability is presented, a methodology for survivable system design is defined, and an illustrative example is presented.

When a natural disaster strikes, a corporation normally places a disaster recovery plan into effect. These plans define how a corporate knowledge base is reconstituted after a catastrophic failure, allowing an enterprise to continue its daily functions. However, natural disasters are relatively rare occurrences. A corporation that leases space at a site hosting facility and purchases disruption insurance has allocated assets in advance, with potentially no return on those investments if a disaster does not occur [1]. In this regard, disaster recovery is like insurance.

With the ubiquity of the Internet, it has become more difficult to disrupt services for an extended period of time. Consumers expect 24-hour service or they take their Internet shopping elsewhere. Global enterprises now link what were isolated data centers to Enterprise Resource Planning systems to manage inventory and track consumer preferences. There is no downtime allowed in today's global economy.

Enter the concept of *survivable information systems*. Survivable information systems continue operating in various failure scenarios, although potentially in a degraded mode. Problems such as denial of service attacks, loss of a local network segment, or loss of a single data center are addressed by using various countermeasures or mechanisms to ensure continued system operation. When a traditional information system is subjected to failure conditions, it shuts down. By contrast, a survivable system continues functioning in support of the enterprise.

This article begins with a discussion of survivability, and contrasts survivability with the traditional disaster recovery and business continuity disciplines. A system survivability design methodology is presented that incorporates risk assessment and risk mitigation activities into the system development life cycle. Finally, representative examples of survivability mechanisms within the context of service-oriented system architectures are presented to assist those designing survivable information systems.

Survivability Defined

The Software Engineering Institute has conducted a comprehensive project on survivable information systems [2]. Survivability has been defined as "the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents." A mission is a set of high-level requirements that a system must fulfill to be considered successful. An organization may not have a mission statement as such, but every organization has some sort of vision that articulates the organization's ambitions.

Figure 1: *Survivability Versus Disaster Recovery Event Categories*

	Predictable	Unpredictable
Controllable	• Develop standard operating procedures	• Quick Response Team • Event Drills • Capture lessons learned
Uncontrollable	• Develop predictive models • Use indicators for avoidance	• Contingency planning

For example, if an electronic commerce server has an expectation of seven days a week, 24 hours a day availability, an unrecoverable disk crash that requires several hours to restore would be considered a failure to fulfill the mission. A *failure* is a potentially damaging event caused by a deficiency in the system or in an external element on which the system depends. An *accident* is defined as a randomly occurring and potentially damaging event such as a natural disaster that is thought of as externally generated. An *attack* is a potentially damaging event caused by an intelligent adversary.

What matters in the context of survivability is not so much the cause of a problem, but the system's response to that problem. To continue functioning, a system must respond to a failure, accident, or attack before the cause can be determined. That is, the system must react to the event, recover from it, and continue its mission. A classic example of a survivable system would be HAL, the all-knowing information system from "2001: A Space Odyssey" [3]. HAL reacted to attempted shut-down operations by protecting itself at the unfortunate expense of the Discovery's crew.

Hardy [4] categorizes events into four quadrants, see Figure 1. Survivable systems address the events that occur in all four quadrants. Events are categorized as either controllable or beyond the control of the system (uncontrollable), and predictable or unpredictable. Events that are predictable and controllable can be scheduled, or monitored and addressed before they become crisis. Unpredictable but controllable events can be addressed within the context of an incident response team such as those used to control malicious code attacks [5]. In a given situation, a survivable system reacts to all types of events and continues operation to fulfill its mission.

In Swanson [6], the notion of disaster

recovery is presented within the context of a major, catastrophic system failure that is caused by an external event and denies access to a normally used facility for an extended period of time. An example would be a hurricane or 100-year flood that decimates a building and the power, water, and transportation infrastructure required for support personnel. The goal of a disaster recovery plan is to allow resumption of operations as quickly as possible, often at an alternate site.

In Hardy's quadrant model, events that are unpredictable and uncontrollable cause activation of the disaster recovery plan. The concept of survivability incorporates disaster recovery, but extends the concept to include *all* events that may disrupt system operations.

Survivability Versus Continuity

Quirchmayr [7] and Nemzow [1] both distinguish the concept of continuity planning from the concept of disaster recovery. These authors consider continuity planning to involve the entire collection of personnel, processes, and procedures that, together with the computing assets, allow a business to continue functioning. In their respective models, the workflow associated with a system is considered at least as critical to an enterprise's survival as the actual hardware and software used to support the personnel.

In these models, action plans incorporate personnel considerations for extended-term outages. For example, if an enterprise's facility is without power, a continuity model would not only address auxiliary uninterruptible power supplies but also address the logistics of maintaining the fuel source, providing support to operations personnel, and providing alternate communications paths to other locations if necessary. Continuity planning integrates the entire business process model into the reaction and recovery tasks associated with enterprise information system operations.

The Goal of Survivability Planning

In an optimal situation, an enterprise can define an acceptable balance between the potential risk of a failure that would render a system unable to fulfill its mission and the cost associated with protective mechanisms. Bakry [8] proposes using economic analysis to optimize the cost of prevention versus risk of failure tradeoff, and the concept of a balanced solution is introduced.

This model is illustrated in Figure 2. For example, if an organization is extremely risk adverse, it can expend considerable assets on redundant computing environments and an alternate support staff. In contrast, an organization that believes a catastrophe will never happen to them might fulfill their plan with a package of writeable DVDs for media backup. In this instance, the organization accepts the risk of system loss in exchange for the cost savings associated with survivable safeguards

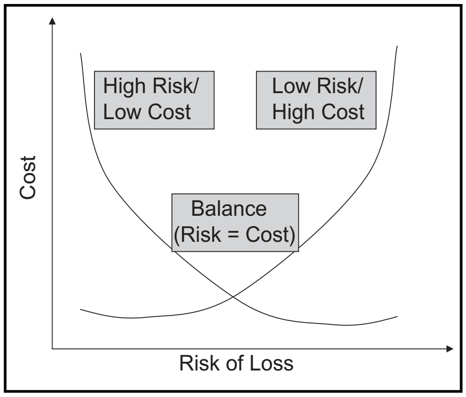


Figure 2: *Risk of Loss Versus Cost of Survivable Safeguard*

A Survivable Design Development Methodology

To improve organizational integration, information systems are becoming increasingly networked to trading partners, customers, and suppliers as well as other sites on the corporate network. In this model, *islands of automation* have been integrated into a network-enabled enterprise benefiting from their interconnectivity. Unfortunately, there is a downside to this model: A company may not have any knowledge about a virus attack that is spreading through a trading partner's network. In such an environment, a system architected with some degree of survivability such as firewalls, virus scanning, or intrusion detection/prevention appliances is most likely to fulfill its mission objectives.

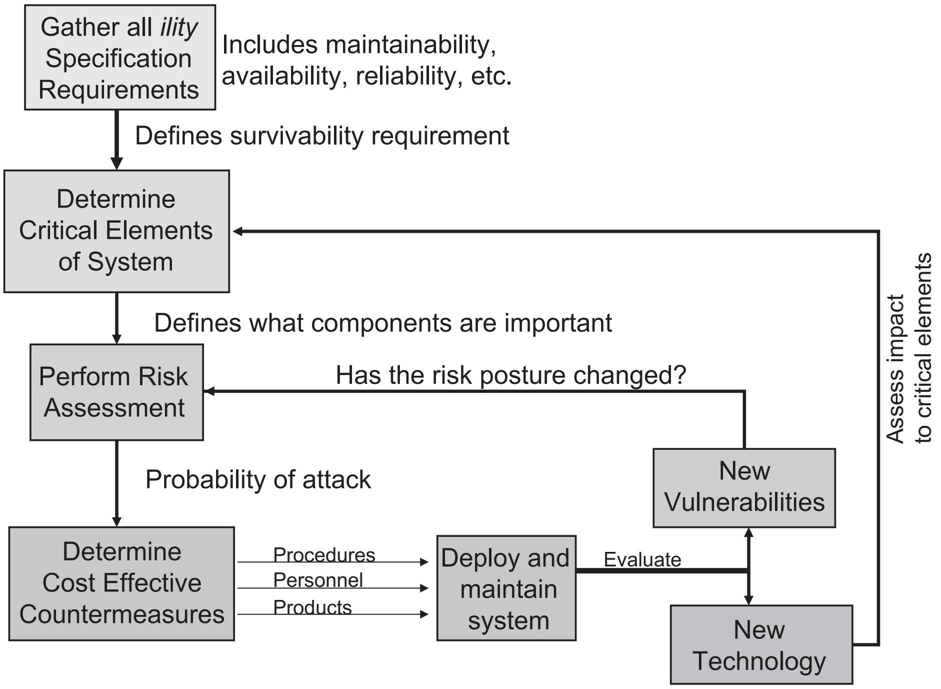
How, then, does a system owner specify or design a system with survivability in mind? There are no generally accepted design methodologies in place to address the diverse events that can impact system survivability. Figure 3 illustrates a methodology that facilitates the integration of survivability characteristics within a traditional information system development framework.

Gathering Survivability Related Requirements

To completely understand the survivability characteristics of an information system, it is essential to understand the system's requirements. In the language of system specification, survivability attributes are usually expressed in terms of specialty engineering disciplines and their requirements. These disciplines include availability, reliability, maintainability, and accountability as well as security and integrity – in Figure 3 they are collected as the *ility* requirements. Table 1 (see page 8) illustrates representative requirements in each of these disciplines that would impact the survivability characteristics of system architecture.

It is important to note that not all survivability constraints require technology-

Figure 3: *Survivability Integration Methodology*


Gather all ility Specification Requirements
Includes maintainability, availability, reliability, etc.
Defines survivability requirement
Determine Critical Elements of System
Defines what components are important
Perform Risk Assessment
Has the risk posture changed?
Probability of attack
Determine Cost Effective Countermeasures
Procedures
Personnel
Products
Deploy and maintain system
Evaluate
New Vulnerabilities
New Technology
Assess impact to critical elements

Requirement Family	Typical requirement statement	Potential survivability impact
Availability	The system shall be available for processing .9995 percent of the time.	Redundant hardware or dedicated communications paths.
Reliability	The system shall have less than 1 hour total downtime per year.	Hardware selection, or need for distributed architecture.
Maintainability	The system components shall be field-replaceable.	On-site replacement parts.
Integrity	The system shall protect information in transit from possible modification.	Secure hash or cryptographic sealing techniques.
Security	The system shall protect information at rest, during processing, and in transit.	Virtual private network (VPN) technology, disk encryption.

Table 1: *Requirement Families That Impact System Survivability*

based solutions. For example, racks containing computer hardware may be sealed with colored tape. If the seal is broken, maintenance personnel would inventory the components and, if necessary, run system diagnostics to detect potential system modifications. Similarly, if unauthorized access to a facility is a concern, a physical security policy that visitors must be escorted at all times provides an acceptable solution.

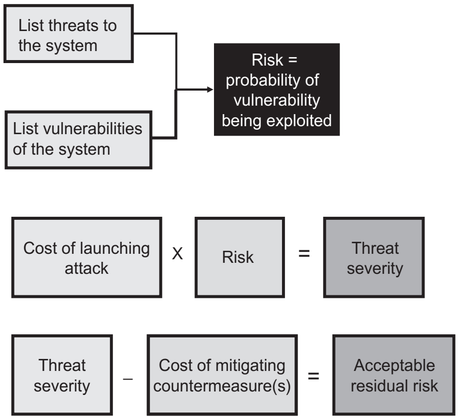
Determine Critical System Elements

From the survivability analysis, critical system elements should be identifiable. Criticality of elements may be based on connectivity requirements, processing capacity, or amount of data accessed. The objective of this process is to determine those elements of the system design whose failure or compromise would have the greatest impact on the operational system. From this decomposition, it is possible to determine which system components will require added care going forward in the development process.

Perform Risk Analysis

At this point, a risk analysis is required. This is not the traditional risk analysis that addresses risk to cost and schedule, but is a risk analysis that assesses the potential impact to the survivability posture of the system. Once the critical elements have been identified, the potential impact of adverse events must be evaluated. Figure 4, adopted from Panko's discussion [9], illustrates the traditional risk assessment process. In risk assessment, the potential threats to the system are enumerated. These threats are then evaluated in the context of system vulnerabilities. That is, a threat to a critical system element is only a threat if the opportunity to exploit a given vulnerability or group of vulnerabilities is present. Whitson [10] presents a basic overview of risk assessment.

Figure 4: *Traditional Risk Assessment Process*


List threats to the system
List vulnerabilities of the system
Risk = probability of vulnerability being exploited
Cost of launching attack
x
Risk
=
Threat severity
Threat severity
–
Cost of mitigating countermeasure(s)
=
Acceptable residual risk

Beyond determining the vulnerabilities, there is the cost associated with exploitation. For example, a risk of data tampering when information is only valid for less than a minute may carry a prohibitively high cost of exploitation. The cost of launching an attack coupled with the risk determines the threat severity. If information is updated every minute, an attacker would have to maintain an alternate data set of sufficient size to hide his intent until the attack is over. In such a case, the perceived cost of the attack would be relatively high, and the risk of detecting fraudulent data would also be relatively high. In such an instant, the threat severity, or consequences, if an attack was launched would be high for the simple fact that a deliberate, concerted attacker would be involved. When threat severity is evaluated in the context of countermeasures, the residual risk associated with system use is derived.

For example, a system that connects to the Internet may have a relatively high risk and a high threat severity. However, using a packet filtering firewall, a minimal set of network services required for the application, and *hardened* or security-conscious host configurations mitigate a considerable amount of risk. Risk to such a system could be further mitigated by using anti-virus software and/or intrusion prevention technology.

The relative costs of architectural elements are significant inputs to the risk assessment process and impact the risk calculation. It should also be noted that some countermeasures may not be intuitively obvious. Creativity and innovation sometimes result in effective solutions for a given enterprise environment.

Countermeasures should also be cost-effective. Not all countermeasures are electronic and computer-intensive. Countermeasures can include standard operating procedures and policies. For example, if unauthorized facility access is a high-risk item, a cost-effective countermeasure could be limiting computer room access to authorized personnel by applying access limiting devices (i.e., locks with keys or smart cards). A guard dog turned loose at night can be just as effective as a sophisticated electronic alarm system.

Deploy and Maintain System

Once the countermeasures have been identified and deployed, the system must be maintained in a survivable state. For example, a system that depends upon anti-virus software must have the latest malicious code signature files downloaded when available. Application updates, or patches, must be tested for compatibility with the application environment and deployed across the enterprise. The best countermeasures in the world do not work if they are not maintained and enforced.

As the system matures, it must be continuously evaluated. For example, all the vulnerabilities associated with a given commercial product may not be applicable to a specific application of the system. A system may not use a given network service, so a patch deployment can be deferred. A new, improved version of a commercial off-the-shelf software component may become available, providing additional functionality without custom software development. In both cases, the relative risks associated with updating the system's architecture must be weighed against the potential risks that could be introduced into the survivability posture.

Threats are continuously evolving as new vulnerabilities and exploits are identified. The risk assessment activity is an ongoing part of the system life cycle. The residual risk associated with the system's survivability posture must be updated on a regular basis to reflect the current system architecture and the current threat environment. A survivability assessment that does not reflect the current state of the system architecture is not a useful document and may inaccurately reflect the risk posture of the system.

Putting the Model to Work

The survivability model described above

has been applied to several architecture development efforts, reflecting diverse environments and their unique operational characteristics. This section discusses how the model was applied to three specific cases.

The first example is a high throughput transaction-processing application. This particular system development was a *renovation* of an existing system that, while still performing adequately, was rapidly becoming unreliable. The fact that a data warehouse hardware platform had reached its end of life made modernization imperative. On first investigation, a high throughput distributed client-server model might have served the purpose. Unfortunately, data replication across the environment could not be supported with the desired reliability (five minutes of downtime per year), and the system maintained highly confidential legal information (criminal records). When the reliability requirements were factored into the equation, a high capacity centralized data warehouse environment with internal transaction process monitoring was a more efficient architecture.

For the second example, a mission-critical networking infrastructure was under consolidation and modernization. The existing network evolved from a series of stovepipe requirements, with each project managing and ordering its own network services. While this approach had served the organization well in the past, it was no longer cost effective or survivable in today's telecommunications environment.

A prioritization of services was undertaken by the customer, moving the network to a reliability-, maintainability-, and availability-based service model. For example, network services that required redundant connectivity and minimal downtime were segregated from traditional administrative-based services that could adapt to a next-business-day restoration. The net result: the organization has been able to reduce the number of redundant communications paths between facilities, reduce costs, and improve management visibility into critical services. Spare components are pre-positioned at strategically placed depot installations instead of stored at every site location.

The third example is a network information resource, responsible for routing user requests to the most probable source of the requested information. In this application, a user's clearance level, bandwidth, and intended use of the information are factored into satisfying the user's query. The application in question has applied distributed client/server architecture used to localize the data storage to its most logical requesters. For example, Pacific Command analysts do not normally explore European data sets. Data has been distributed to the most likely user base with replicated backup services on other servers.

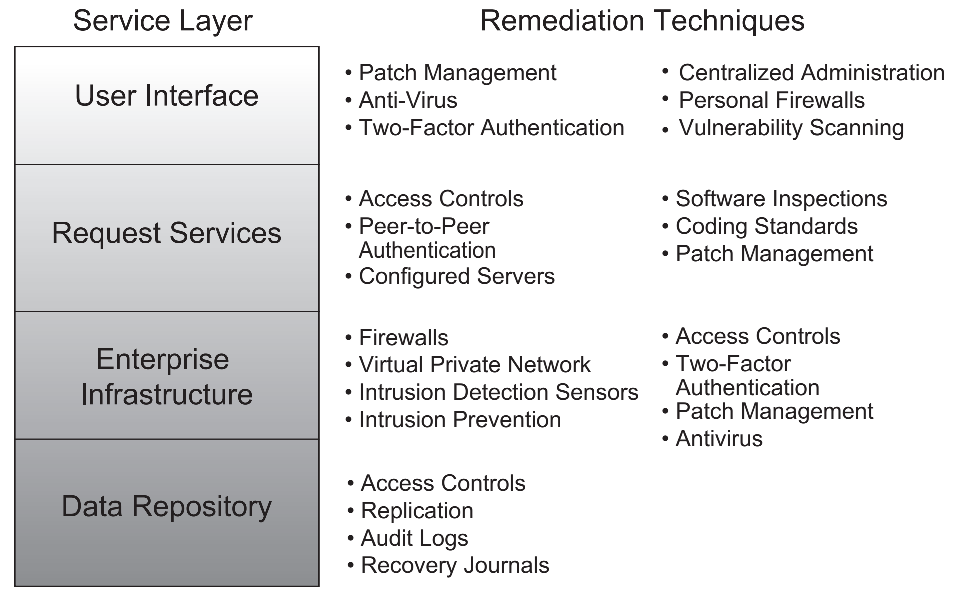


Figure 5: *Service-Oriented Architecture With Remediation Techniques*

> "Once the countermeasures have been identified and deployed, the system must be maintained in a survivable state ...The best countermeasures in the world do not work if they are not maintained and enforced."

A Service-Oriented Example for the Future

Service-oriented architectures (SOAs) decouple data from both the user and the processing services in a layered structure. The most critical data in a SOA may be the processing required to fulfill a user request, or it may be the data repository that contains information vital to the enterprise's mission. Figure 5 illustrates a representative SOA with illustrative remediation techniques that could be applied at each layer.

In most enterprises, the user interfaces are assumed to execute on a standard enterprise desktop system (i.e., Linux, Windows, or Macintosh). Each enterprise configures and manages their desktops differently, depending on the information technology budget and capabilities of the end users. As a result, best practice desktop computing practices are usually applied such as virus scanning and/or patch management. Due to the multi-purpose nature of user desktops, they are considered high risk for introducing potential vulnerabilities into the enterprise. Backup media are usually the responsibility of the end user, unless the enterprise provides global backup services.

The request services layer may represent a significant investment for the enterprise. This layer usually addresses pre- and post-processing needed to make the data meaningful to the user's presentation environment. For example, an application that contains a corporate knowledge base or expert system may represent irreplaceable domain expertise. This type of application can be applied as an analyst's aid to filter large data sets. In these cases, survivability can be enhanced through using good software development and maintenance practices, including configuration management and code escrow.

The enterprise infrastructure is the lifeblood of a services-based architecture. Without the enterprise communication services, data cannot be moved across the layers of the processing architecture in response to user requests. Because the enterprise infrastructure is responsible for both the availability of the information and the integrity of the data in transit, it represents a significant risk to the survivability posture of the enterprise. This is the reason most modern enterprise infrastructures are well protected. Mechanisms employed at the infrastructure layer include virtual private networks, intrusion detection sensors, and firewalls. Each enterprise determines the residual risk associated with the infrastructure and defines appropriate countermeasures as required for the applications it supports.

The data repository may represent the most critical portion of the application. It could contain the corporation's financial records, business intelligence information, or customer records. This information, frequently gathered over extended periods of time, may be the most irreplaceable in an enterprise. As such, the data repository is usually subject to layered data integrity and defense mechanisms. These may include replicating updates to alternate geographically dispersed sites, using journaling and recovery to ensure transactions are saved to the database completely, and using additional authentication techniques for database restructuring. The repository contains the data that allows the applications to perform their analytical or reporting functions, and is vital to the extended life of the enterprise.

Conclusion

The countermeasures in the simple example described above are representative of the types of mechanisms that can be deployed to enhance the survivability of an information system. However, no countermeasure should be deployed without completion of a cost/benefit analysis. There are times when a very elegant, 100 percent effective countermeasure is not the best fit for an enterprise: when deployment would require major upgrades to other portions of the information technology environment, or substantively damage the functionality of existing applications. The objective is to make an enterprise information system survivable, not inaccessible.

Survivability can be attained by augmenting the traditional system development paradigms with a relatively small set of process augmentations. The goal is creation of a risk-oriented model of the system that allows the owner/creator to make sound decisions about design alternatives impacting the survivability of the system. When such models are in place, they can effectively enhance the survivability posture of the system. In such environments, expenditures on major disaster recovery plans can be greatly reduced because the system has integrated reaction and recovery capabilities.

Survivability as a design consideration yields a more effective return on investment than expenditures for redundant hardware, hot site backup, and major disaster preparedness measures. The incorporation of survivability addresses all potential disaster scenarios, not just the most catastrophic, resulting in a more prepared organization that can react effectively to multiple event scenarios, in effect creating a more adaptable and agile system.◆

Information Sources

For additional information on survivability, the Software Engineering Institute at <www.sei.cmu.edu> has ongoing projects on survivable design, and runs the U.S. Computer Emergency Response Team (U.S.-CERT[®]) Coordination Center <www.us-cert.gov>. For additional information on contingency planning and disaster recovery, the National Institute of Standards and Technology hosts a Computer Security Resource Clearinghouse at <www.csrc.nist.gov>. The clearinghouse includes a selection of template documents for disaster recovery contributed by various federal chief information officers as best standard practices for large enterprises.

References

1. Nemzow, M. "Business Continuity Planning." International Journal of Network Management 7 (July 1997): 127-136.
2. Ellison, R.J., D.A. Fisher, R.C. Linger, H.F. Lipson, T. Longstaff, and N.R. Mead. "Survivable Network Systems: An Emerging Discipline." Pittsburgh, PA: Software Engineering Institute, 1999: 1-3 <www.sei.cmu.org>.
3. Clarke, A.C. 2001: A Space Odyssey. Reissue ed. New York: Roc, Sept. 2000.
4. Hardy, K. "Contingency Planning." Business Quarterly 56.4 (1992): 26-28.
5. Allen, J.H. The CERT Guide to System and Network Security Practices. 1st ed. Upper Saddle River, NJ: Addison-Wesley, 2001: 447.
6. Swanson, M., et al. Contingency Planning Guide for Information Technology Systems: Recommendations of the National Institute of Standards and Technology. Diane Pub. Co., May 2004 <www.nist.gov>.
7. Quirchmayr, G. Survivability and Business Continuity Management. Eds. P. Montague and C. Steketee. Proc. of the Second Australasian Information Security Workshop, Dunedin, New Zealand, Jan. 2004.
8. Bakry, S.H. "Development of Security Policies for Private Networks." International Journal of Network Management 13 (2003): 203-120.
9. Panko, R.R. Corporate Computer and Network Security. 1st ed. Upper Saddle River, NJ: Pearson Education, Inc., 2004.
10. Whitson, G. "Computer Security: Theory, Process, and Management." Consortium for Computing Sciences in Colleges 2003.

About the Author

Ronda R. Henning is a senior scientist in the Government Communications Systems Division at Harris Corporation, an international communications company. She is the network security manager for the Federal Aviation Administration's Telecommunication Infrastructure program, a $1.7 billion network modernization of the National Air Space. Previously, she led the Harris Information Assurance Center of Excellence in defining security architectures for the National Crime Information Center and the Eastern Test Range Modernization Programs. Henning also served as principal investigator on the Network Vulnerability Visualization Architecture program, and the Integrated Design Environment for Assurance program. Prior to this, she worked in information security research and development at the National Security Agency. She is a Certified Information Systems Security Professional and a Certified Information Security Manager. Henning has a Masters of Business Administration from the Florida Institute of Technology, a Master of Science in computer science from Johns Hopkins University, and a Bachelor of Arts from the University of Pittsburgh.

Harris Corporation
Government Communications
Systems Division
MS F-11
1025 W NASA BLVD
Melbourne, FL 32919
Phone: (321) 309-2642
Fax: (321) 309-2590
E-mail: rhenning@harris.com

[®] CERT is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.