SURVIVABILITY: Protecting Your Critical Systems

ROBERT J. ELLISON, DAVID A. FISHER, RICHARD C. LINGER, HOWARD F. LIPSON, THOMAS A. LONGSTAFF, AND NANCY R. MEAD
CERT Coordination Center, Software Engineering Institute

Society is increasingly dependent upon large-scale, distributed systems that operate in unbounded network environments. Survivability helps ensure that such systems deliver essential services and maintain essential properties in the face of attacks, failures, and accidents.

Contemporary large-scale distributed networks are being used to achieve radical new levels of organizational integration. This integration obliterates traditional organizational boundaries and ties local operations into components of comprehensive, network-based business processes. For example, commercial organizations are integrating operations with business units, suppliers, and customers through large-scale networks that enhance communication and services. These networks combine previously fragmented operations into coherent processes open to many organizational participants. This new paradigm represents a shift from bounded networks with central control to unbounded networks, where administrative control is distributed without central authority (see the sidebar "Glossary of Survivability Terms" on page 56).

Organizational integration is accompanied by elevated risks of intrusion and compromise. These risks can be mitigated by incorporating survivability capabilities into an organization's systems. *Survivability* is the capability of a system to fulfill its mission in a timely manner in the presence of attacks, failures, or accidents. The emphasis of survivability is on continuity of operations, with the understanding that security precautions cannot guarantee that systems will not be penetrated and compromised. Survivability focuses on unbounded networked systems where traditional security measures are inadequate. As an emerging discipline, it builds on related fields of study (such as security, fault tolerance, reliability, and verification) and introduces new concepts and principles.

MISSION FULFILLMENT

In survivability engineering, it is the fulfillment of a mission that must survive an attack, not any particular subsystem or system component. A *mission* is a set of very high level requirements or goals. Missions are not limited to military settings; any successful organization or project must have a vision

1089-7801/99/$10.00 ©1999 IEEE

GLOSSARY OF SURVIVABILITY TERMS

Accidents—a broad range of randomly occurring and potentially damaging events such as natural disasters. Accidents are often externally generated events.

Adaptation services—system functions provided to continually improve a system's capability to deliver essential services, typically by improving resistance, recognition, and recovery capabilities.

Attack—a series of steps taken by an intelligent adversary to achieve an unauthorized result. Attacks include intrusions, probes, and denials of service.

Essential services—services that must be provided to system users even in the presence of attacks, failures, or accidents.

Failure—a potentially damaging event caused by deficiencies in the system or in an external element on which the system depends. Failures may be due to software design errors, hardware degradation, human errors, or corrupted data.

Recognition services—system functions that detect attacks and the extent of system damage or compromise.

Recovery services—system functions to support the restoration of services after an attack has occurred. Recovery services also help a system maintain essential services during an attack.

Resistance services—system functions that repel attacks and make them difficult and costly.

Survivability—a system's capability to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents.

Unbounded network—computer system or systems characterized by distributed administrative control without central authority, limited visibility beyond the boundaries of local administration, and lack of complete information about the network.

of its objectives, whether they are expressed implicitly or as a formal mission statement. Judgments as to whether or not a mission has been fulfilled are typically made in the context of external conditions that may affect the achievement of that mission's goals. Timeliness is typically included in (or implied by) the very high level requirements that define a mission. However, timeliness is such an important factor that it is explicit in the definition of survivability.

Assume that a financial system shuts down for 12 hours during a period of widespread power outages caused by a hurricane. If the system preserves the integrity and confidentiality of its data and resumes its essential services after the period of environmental stress is over, the system can reasonably be judged to have fulfilled its mission. However, if the same system shuts down unexpectedly for 12 hours under normal conditions (or under relatively minor environmental stress) and deprives its users of essential financial services, the system can reasonably be judged to have failed its mission, even if data integrity and confidentiality are preserved.

The terms *attack*, *failure*, and *accident* include all potentially damaging events, but do not partition these events into mutually exclusive or even distinguishable sets. It is often difficult to determine if a particular detrimental event is the result of a malicious attack, a component failure, or an accident. Even if the cause is eventually determined, the immediate response cannot depend on speculations about the cause.

- Attacks include intrusions, probes, and denials of service orchestrated by an intelligent adversary. The mere threat of an attack can have as severe an impact on a system as an actual occurrence. A system that assumes an overly defensive position because of the threat of an attack may significantly reduce its functionality by diverting excessive resources to monitoring the environment and protecting system assets.
- Failures are the result of deficiencies in the system or in an external element on which the system depends. Failures may be due to software design errors, hardware degradation, human errors, or corrupted data.
- Accidents describe a broad range of randomly occurring and potentially damaging events such as natural disasters. Accidents are often externally generated events whereas failures are typically internally generated events.

With respect to system survivability, distinctions between attacks, failures, and accidents are less important than the event's impact. Our survivability approach concentrates on the effect of a potentially damaging event. Typically, for a system to survive, it must react to (and recover from) a damaging effect (for example, the integrity of a database is compromised) long before the underlying cause is identified. In fact, the reaction and recovery must be successful whether or not the cause is ever determined.

SURVIVABILITY IN UNBOUNDED NETWORKS

The success of a survivable system depends on the computing environment in which it operates. The trend in networked computing environments is toward largely unbounded network infrastructures. A bounded system is one in which all of the system's parts are controlled by a unified administration and can be completely characterized and controlled. At least in theory, the behavior of a bounded system can be understood and all of its various parts identified.

In an unbounded system there is no unified administrative control over the system's parts. The term *administrative control* is used here in the strictest sense: It includes the power to impose and enforce sanctions and not simply to recommend an appropriate security policy. In an unbounded system, each participant has an incomplete view of the whole, must depend on and trust information supplied by its neighbors, and cannot exercise control outside its local domain.

An unbounded environment exhibits the following properties:

- It encompasses multiple administrative domains with no central authority.
- It lacks global visibility (that is, the number and nature of the nodes in the network cannot be fully known).
- Interoperability between administrative domains is determined by convention.
- Systems are widely distributed and interoperable.
- Users and attackers can be peers in the environment.
- It cannot be partitioned into a finite number of bounded environments.

An unbounded system can be composed of bounded and unbounded systems connected together in a network. Although the security policy of an individual bounded system cannot be fully enforced outside the boundaries of its administrative control, the policy can be used as a yardstick to evaluate the security state of that bounded system. Of course, the security policy can be advertised outside the bounded system, but administrators are severely limited in their ability to compel or persuade outside individuals or entities to follow it. This limitation is particularly true when an unbounded domain spans jurisdictional boundaries, making legal sanctions difficult or impossible to impose.

Survivability on the Internet

The Internet is an example of an unbounded environment with many client-server network applications. Lack of central administrative control and of global visibility characterizes the Internet and the distributed applications residing on it. A public Web server and its clients may exist within many different administrative domains on the Internet.

More and more, a company's partners on one project are its competitors on the next, making trust an extremely complex concept.

Many business-to-business Web-based e-commerce applications depend on conventions within a specific industry segment for interoperability. There is little distinction between insiders and outsiders—anyone connected to the Internet is an insider, whether or not they are known to a particular subsystem. This characteristic is the result of the desire, and modern necessity, for connectivity. A company cannot survive in a highly competitive industry without easy and rapid access to its customers, suppliers, and partners.

More and more, a company's partners on one project are its competitors on the next, making trust an extremely complex concept. Trust relationships are continually changing and, in traditional terms, may be highly ambiguous. Trust is especially difficult to establish in the presence of unknown users from unknown sources outside a company's administrative control. Legitimate users and attackers are peers in the environment and there is no method to isolate one group from the other. In other words, there is no way to bound the environment to legitimate users solely through a common administrative policy.

Security-Based Defense

Most security technology depends on certain underlying assumptions about the nature and structure of systems.[1,2] Generally, these assumptions include closed systems with central administrative control and the capability to observe any desired activity within the system—assumptions that may have been appropriate when systems were isolated islands with highly controlled interfaces.

Today, however, systems are open, with no one person or organization having administrative control, and with any observer—whether inside or outside the system—having only limited visibility into the structure, extent, and topology of the system.

Much of today's research and practice in computer system survivability takes a security-based view of defense against computer attacks. The traditional firewall concept[3] has been expanded into what are called *boundary controllers*. For example, a secure Department of Defense domain might use commercial and nonsecure products for general-purpose computing, with boundary controllers such as the Naval Research Laboratory pump[4] moving data among domains with differing security policies. The Java security model,[5] in particular the sandbox, applies a similar kind of isolation to imported Java components so that their functionality can be limited to maintain a secure environment.

The overall function of a system should adapt to preserve essential services.

For survivability, this kind of approach is incomplete because it focuses almost exclusively on prevention (that is, hardening a system to prevent a break-in or other malicious attack). It does little to help an organization detect an attack or recover after a successful attack has occurred. This security-focused view is also limited by evaluation techniques that concentrate on the relative hardness of a system—as opposed to a system's robustness under attack, its ability to recover compromised capabilities, or its ability to function correctly in the presence of compromised components.

Affordability and COTS Components

Affordability is always a significant factor in the design, implementation, and maintenance of systems, and it encourages sharing and replication of components. That sharing extends to the national infrastructure (for example, the power grid, the public switched communications networks, and the financial networks) and to national defense. In fact, the trend toward increased sharing of common infrastructure components in the interest of economy virtually ensures that the civilian networked information infrastructure and its vulnerabilities will always be an inseparable part of a country's national defense.[6]

Practical, affordable systems are almost never completely custom-built, but rather are constructed from commonly available commercial off-the-shelf (COTS) components. The trend toward developing systems through integration and reuse rather than customized design and coding is a cornerstone of modern software engineering. Unfortunately, the intellectual complexity associated with software design, coding, and testing virtually ensures the presence of bugs in COTS components that can be exploited by attackers. These bugs can and will be discovered in commercial and public-domain products whose internal structures are widely available and hence can be analyzed by those who wish to exploit the weaknesses. When these products are incorporated as components of larger systems, those systems become vulnerable to attack strategies based on the exploitable bugs, making it possible for a single-attack strategy to have a wide-ranging and devastating impact.

SURVIVABILITY OF ESSENTIAL SERVICES AND PROPERTIES

Key to the concept of survivability is the identification of essential services, and the essential properties that support them, within an operational system. *Essential services* are defined as the functions of the system that must be maintained to meet the mission requirements when the environment is hostile, or when failures or accidents occur that threaten the system.

To maintain their capability to deliver essential services, survivable systems must exhibit four key properties (see Table 1):

- resistance to attacks,
- recognition of attacks,
- full recovery of essential services after attack, and
- adaptation and evolution to reduce effectiveness of future attacks.

There are typically many services that can be temporarily suspended while a system deals with an attack or other extraordinary environmental condition. Such a suspension can help isolate areas that have been affected by an intrusion and can free up system resources to deal with the intrusion's effects. The overall function of a system should adapt to preserve essential services.

Table 1. A survivable system must exhibit four key properties.

Property	Description	Examples
Resistance to attacks	Strategies for repelling attacks	System and user authentication, access control, encryption, firewalls, proxy servers, strong configuration management, dispersion of data, diversification of systems, application of system upgrades for known vulnerabilities
Recognition of attacks and the extent of damage	Strategies for detecting attacks (including intrusions) and understanding the current state of the system, including evaluating the extent of damage	Recognition of intrusion usage patterns, virus scans, internal integrity checking, auditing, system configuration monitoring, and network monitoring
Recovery of full and essential services after attack	Strategies for restoring compromised information or functionality, limiting the extent of damage, maintaining or restoring essential services within the time constraints of the mission, restoring full service as conditions permit	Restoration of data and programs, use of alternative services, use of redundant modules with the same interface but different implementation, operational procedures to restore system configurations, isolation of damage, ability to operate with reduced services or reduced user community
Adaptation and evolution to reduce effectiveness of future attacks	Strategies for improving system survivability based on knowledge gained from attacks	Incorporation of new patterns for intrusion recognition, adaptive filtering, and logging

Central to the delivery of essential services is the capability of a system to maintain *essential properties* (that is, specified levels of quality attributes such as integrity, confidentiality, and performance). Thus, it is important to define minimum levels of quality attributes that must be associated with essential services. For example, a launch of a missile by a defensive system cannot be effective if the system's performance is slowed to the point that the target is out of range before the system can launch.

The capability to deliver essential services (and maintain the associated essential properties) must be sustained even if a significant portion of the system is incapacitated. Furthermore, this capability should not be dependent upon the survival of a specific information resource, computation, or communication link. In a military setting, essential services might be those required to maintain an overwhelming technical superiority, and essential properties may include integrity, confidentiality, and a level of performance sufficient to deliver results in less than one decision cycle of the enemy. In the public sector, a survivable financial system is one that maintains the integrity, confidentiality, and availability of essential information and financial services, even if particular nodes or communication links are incapacitated because of an intrusion, failure, or accident, and that recovers compromised information and services in a timely manner. The financial system's survivability might be judged using a composite measure of the disruption of stock trades or bank transactions (that is, a measure of the disruption of essential services).

Again, ultimately, it is mission fulfillment that must survive, not any portion or component of the system. A lost essential service can be replaced by another service that supports mission fulfillment in a different but equivalent way. However, we still believe that the identification and protection of essential services is an important part of a practical approach to building and analyzing survivable systems. Thus in our definition of essential services, we include alternate sets of essential services (perhaps mutually exclusive) that need not be simultaneously available. For example, a set of essential services to support power delivery may include both the distribution of electricity and the operation of a natural gas pipeline.

SURVIVABILITY SOLUTIONS

Survivability solutions are best understood as risk-management strategies that depend first on an inti-

mate knowledge of the mission being protected.[2] The mission focus expands survivability solutions beyond purely independent ("one size fits all") technical solutions, even if those technical solutions extend beyond traditional computer security to include fault tolerance, reliability, usability, and so forth. Risk-mitigation strategies must be created in the context of a mission's requirements (prioritized sets of normal and stress requirements), and must be based on "what-if" analyses of survival scenarios and contingency planning. Only then can we look toward generic software engineering solutions based on computer security, other software quality attribute analyses, or other strictly technical approaches to support the risk-mitigation strategies.

The preparatory steps necessary for survivability must be taken by an organization as a whole.

To reduce the combinatorics inherent in creating representative sets of survival scenarios, the scenarios must focus on adverse effects rather than causes. Effects are also more important than causes in the immediate situation, because an organization will likely have to deal with (and survive!) an adverse effect long before a determination is made as to whether the cause was an attack, a failure, or an accident. Awaiting the outcome of a detailed postmortem to determine the cause before acting to mitigate the effect is out of the question for most modern, mission-critical applications.

Contingency (including disaster) planning requires that risk-management decisions and economic trade-offs be made by executive management, with guidance from technical experts in the application domain, computer security, and other software engineering and related disciplines. Survivability depends at least as much upon the risk-management skills of an organization as it does upon the technical expertise of a cadre of computer-security experts. This is certainly appropriate from an organizational perspective, because responsibility for business risk management belongs to executive management, not to computer-security experts or other technical personnel. The role of the experts in security, the application domain, and other technically relevant areas is to provide executive management with the information necessary to make informed risk-management decisions. Thus, the preparatory steps necessary for survivability must be taken by an organization as a whole, rather than by security experts alone.

New Tools for Survivability Support

New research methods and tools to support survivability solutions are under development. A number of these efforts focus on architectural issues.

One approach motivated by information warfare attacks on the U.S. infrastructure proposes to designate a portion of the infrastructure as the essential minimum and harden that portion against attacks. Recent work proposes methodology to analyze that approach.[7]

Neumann documents the first phase of a multiyear effort on survivability.[8] The overall objectives of the project include

- defining survivability requirements,
- identifying functionality to support these requirements,
- exploring techniques for designing and developing highly survivable systems and networks, despite the presence of untrustworthy subsystems and untrustworthy participants, and
- recommending specific architectural structures that can lead to survivable systems and networks capable of either preventing or tolerating a wide range of threats.

Sullivan takes a control systems perspective on survivability.[9] A *control system* manages the behavior of a monitored system within its environment to maintain the acceptable operation of the system. An *adaptive control system* provides control of a system in the face of disruption to elements of the system and its control system.

Thursisingham examines survivability requirements for real-time command and control systems[10] to determine software infrastructure requirements and identify a migration path for legacy systems.

The CERT Coordination Center is developing a Survivable Network Analysis (SNA) method to evaluate the survivability of systems in the context of attack scenarios. Also under development is a Survivable Systems Simulator that will provide for the analysis, testing, and evaluation of survivability solutions in unbounded networks.

The SNA method permits assessment of survivability strategies at the architecture level. Steps in the SNA method include

PROTECTING CRITICAL INFRASTRUCTURES

Much of the research in survivability relates to protecting critical national infrastructures. These infrastructures include the electric power grid (and other energy infrastructures), transportation, telecommunications, health care, banking and finance, and national defense. Particularly in the U.S. and Europe, these infrastructures increasingly rely on large-scale, highly distributed software systems operating over open, unbounded networks. Although this increases the efficiency and sophistication of the services these infrastructures provide, it also increases their vulnerability to cyber-attack.

In response to the U.S. Presidential Commission report on critical infrastructure protection,[1] Presidential Decision Directive 63 (PDD 63)[2] established new government structures, including the National Infrastructure Protection Center and the Critical Infrastructure Assurance Office. The NIPC (http://www.fbi.gov/nipc/welcome.htm) is the U.S. government's focal point for threat assessment, warning, investigation, and response to threats or attacks against critical infrastructures. The CIAO (http://www.info-sec.com/ciao/) is responsible for integrating the various sector plans into a National Infrastructure Assurance Plan and coordinating analyses of the U.S. government's dependencies on critical infrastructures.

The Defense Advanced Research Projects Agency (DARPA) funds ongoing national research in information survivability. Research areas include intrusion detection, intrusion-tolerant systems, barriers, strategic intrusion assessment, and security architectures. Information about this research can be found at http://www.darpa.mil/ito/research/is/ and http://www.darpa.mil/ito/research/tnt/.

The European Dependability Initiative (http://www.cordis.lu/esprit/src/stdepend.htm) represents a major research effort in the European Union to address many of the same issues and concerns as the critical infrastructure protection and survivability efforts in the U.S., and includes plans for joint EU-U.S. collaboration.

The IEEE Computer Society's Technical Committee on Fault-Tolerant Computing and IFIP Working Group 10.4 on Dependable Computing and Fault Tolerance have formed dependability.org (http://www.dependability.org/), a Web resource on the technology of dependable systems.

References

1. *Critical Foundations—Protecting America's Infrastructures*, Report of the Presidential Comm. on Critical Infrastructure Protection, Oct. 1997, p. 173; available online at http://www.pccip.gov/.
2. Presidential Decision Directive 63 (PDD 63), "Protecting America's Critical Infrastructures," http://www.info-sec.com/ciao/63factsheet.html.

Further Reading

Books

- J.C. Lapie, ed., *Dependability: Basic Concepts and Terminology*, Springer-Verlag, New York, 1992.
- N.G. Leveson, *Safeware: System Safety and Computers*, Addison-Wesley, New York, 1995.
- J. Musa et al., *Software Reliability: Measurement, Prediction, and Application*, McGraw-Hill, New York, 1987.
- F.B. Schneider, ed., *Trust in Cyberspace*, National Research Council, Committee on Information Systems Trustworthiness, National Academy Press, Washington, D.C., 1999; also available online at http://www.nap.edu/readingroom/books/trust/.

Proceedings

- R. Kazman et al., "The Architecture Tradeoff Analysis Method," *Proc. IEEE Int'l Conf. Eng. of Complex Computer Systems*, IEEE CS Press, Los Alamitos, Calif., 1998; available online at http://www.sei.cmu.edu/ata/.
- *Proc. 1997 Information Survivability Workshop*, Software Eng. Institute and IEEE CS Press, Los Alamitos, Calif., 1997; available online at http://www.cert.org/research/.
- *Proc. 1998 Information Survivability Workshop*, Software Eng. Institute and IEEE CS Press, 1998; also available online at http://www.cert.org/research/.

Articles and Reports

- C. Ebert, "Dealing with Nonfunctional Requirements in Large Software Systems," *Annals of Software Eng.*, Vol. 3, Sept. 1997, pp. 367–395.
- Reliable Software Technologies research projects, 1999, http://www.rstcorp.com/research/projects.html.
- Trusted Information Systems research projects, 1999, http://www.nai.com/nai_labs/asp_set/intro.asp.

- elicitation of system mission and architecture,
- identification of essential service scenarios and corresponding architecture components,
- generation of attack scenarios and corresponding compromisable architecture components, and
- survivability analysis of architectural components that are both essential and compromisable.

Attack scenarios play a key role in the method. SNA results are summarized in a survivability map that links recommended survivability strategies for

resistance, recognition, and recovery to the system architecture and requirements. The SNA method has been applied to a subsystem of a large-scale, distributed health-care system.[11] Future studies will apply the SNA method to proposed and existing distributed systems for government, defense, and commercial organizations.

The Survivable Systems Simulator being developed by the CERT Coordination Center is based upon a new methodology called "emergent algorithms."[12] Emergent algorithms produce global effects through cooperative local actions distributed throughout a system. These global effects (which "emerge" from local actions) can support system survivability by allowing a system to fulfill its mission, even though the individual nodes of the system are not survivable. Emergent algorithms can provide solutions to survivability problems that cannot be achieved by conventional means. The Survivable Systems Simulator will allow stakeholders to visualize the effects of specific cyber-attacks, accidents, and failures on a given system or infrastructure. The goal is to enable "what-if" analyses and contingency planning based on simulated walk-throughs of survivability scenarios.

Development Considerations

For new systems, survivability imposes constraints on all phases of the software development process.

- At the requirements and specification level, essential services and assets should be identified. Requirements for resistance, recognition, recovery, and adaptation should also be specified.
- Architectures should incorporate survivability strategies such as those mentioned in Table 1. Evaluation should treat survivability on par with other properties such as performance, reliability, and maintainability.
- Reused and COTS products should be selected with survivability in mind.
- Design and implementation should include techniques for isolation, replication, restoration, and migration of essential services.
- Correctness verification should ensure faithful implementation of survivability specifications.
- Testing should assess the reliability of survivability functions operating in cooperation with other system functions in adverse environments.
- Finally, procedures for system operations should have a substantial impact on survivability. They should include processes for managing survivability policies, responding to attacks, and taking recovery actions.

For existing systems, survivability provides a new perspective on evolution and upgrade. The survivability of existing systems can often be improved with additional layers of boundary control—for example, firewalls and their more sophisticated successors—and through evolution to redundant (and diverse) hardware and software environments. In addition, administrative procedures for backup, restoration, and migration can be tested and any inadequacies addressed. And survivability features can play a prominent role in the evaluation and selection of vendors and products.

The natural escalation of offensive threats versus defensive countermeasures has demonstrated time and again that no practical systems can be built that are invulnerable to attack. Despite the industry's best efforts, there can be no assurance that systems will not be breached. Thus, the traditional view of information systems security must be expanded to encompass the specification and design of survivability behavior, enabling the creation of systems that are robust in the presence of attack and are able to survive attacks that cannot be completely repelled. ■

ACKNOWLEDGMENTS

The Software Engineering Institute at Carnegie Mellon University is a federally funded research and development center sponsored by the U.S. Department of Defense. CERT and CERT Coordination Center are registered in the U.S. Patent and Trademark Office.

Further information on this article and related research is available at both http://www.sei.cmu.edu/organization/programs/nss/surv-net-tech.html and http://www.cert.org/research.

REFERENCES

1. B. Blakley, "The Emperor's Old Armor," *Proc. 1996 New Security Paradigms Workshop*, ACM, New York, 1997.
2. H.F. Lipson and D.A. Fisher, "Survivability—A New Technical and Business Perspective on Security," *Proc. 1999 New Security Paradigms Workshop*, ACM Press, New York, 1999.
3. S. Bellovin and W. Cheswich, *Firewalls and Internet Security*, Addison-Wesley, Reading, Mass., 1994.
4. M.H. Kang, A.P. Moore, and I.S. Moskowitz, "Design and Assurance Strategy for the NRL Pump," *Computer*, Vol. 31, No. 4, Apr. 1998, pp. 50-64.
5. G. McGraw and E. Felton, *Java Security*, John Wiley & Sons, New York, 1997.
6. *Critical Foundations—Protecting America's Infrastructures*, Report of the Presidential Comm. on Critical Infrastruc-

ture Protection, Oct. 1997; available online at http://www.pccip.gov.
7. R.H. Anderson et al., "Security of the U.S. Defense Information Infrastructure: A Proposed Approach," Rand Report MR-993-OSD/NSA/DARPA, 1999.
8. P.G. Neumann, "Practical Architectures for Survivable Systems and Networks: Phase-One Final Report," 1998; available online at http://www.csl.sri.com/neumann/arl-one.html.
9. K. Sullivan et al., "Information Survivability Control Systems," *Proc. 1999 Int'l Conf. Software Eng.*, IEEE Computer Society Press, Los Alamitos, Calif., 1999.
10. B.M. Thursisingham and J.A. Maurer, "Information Survivability for Evolvable and Adaptable Real-Time Command and Control Systems," *IEEE Trans. Knowledge and Data*, Vol. 11, No. 1, Jan./Feb. 1999, pp. 228-238.
11. R.J. Ellison et al., "Survivable Network Systems Analysis: A Case Study," *IEEE Software*, Vol. 16, No. 4, July/Aug. 1999, pp. 70-77.
12. D.A. Fisher and H.F. Lipson, "Emergent Algorithms—A New Method for Enhancing Survivability in Unbounded Systems," *Proc. 32nd Ann. Hawaii Int'l Conf. System Sciences* (HICSS-32), IEEE CS Press, Los Alamitos, Calif., 1999; also available online at http://www.cert.org/research/.

Robert J. Ellison is a senior member of the technical staff in the CERT Coordination Center at the Software Engineering Institute. His research interests include system survivability, software development environments, and CASE tools. Ellison received a PhD in mathematics from Purdue University. He is a member of IEEE Computer Society and ACM.

David A. Fisher is a senior member of the technical staff in the CERT Coordination Center at the Software Engineering Institute. He currently leads an effort developing new tools for simulating and visualizing survivability problems and solutions in unbounded networks. He received a PhD in computer science from Carnegie Mellon University.

Richard C. Linger is a senior member of the technical staff in the CERT Coordination Center at the Software Engineering Institute and an adjunct faculty member at the Heinz School of Public Policy and Management, Carnegie Mellon University. His research interests include survivable systems, semantics of network architectures, large-scale system development, and process improvement. He holds a BSEE from Duke University and is a member of IEEE and ACM.

Howard F. Lipson is a senior member of the technical staff in the CERT Coordination Center at the Software Engineering Institute. His research interests include the design and analysis of survivable systems, survivable systems simulation, and critical infrastructure protection. Lipson has chaired two IEEE-sponsored workshops on survivability. He holds a PhD in computer science from Columbia University, and is a member of IEEE and ACM.

Thomas A. Longstaff manages research and development in network security for the CERT Coordination Center at the Software Engineering Institute. His research interests include information survivability and critical national infrastructure protection. Longstaff has a PhD in computer science from the University of California, Davis.

Nancy R. Mead is a senior member of the technical staff in the CERT Coordination Center at the Software Engineering Institute, and a faculty member in the Master of Software Engineering Program, Carnegie Mellon University. Her research interests are software requirements engineering, software architectures and metrics, and real-time systems. Mead received a PhD in mathematics from Polytechnic Institute of New York. She is a senior member of IEEE and IEEE Computer Society, and a member of ACM.

Readers may contact Ellison at the Software Engineering Institute, 4500 Fifth Avenue, Pittsburgh, PA 15213; ellison@sei.cmu.edu.