Dr. Wolthusen is deputy head of the security technology department at Fraunhofer-IGD, a German national research laboratory. He has worked in information security and information assurance for more than a decade, is a member of the task force on information assurance of the Institute of Electrical and Electronics Engineers, and vice chair for Europe of the IEEE Information Assurance Standards Committee. He is the author or coauthor of more than twenty articles and three books and holds a Ph.D. and M.Sc. in computer science from the Technische Universität Darmstadt, Germany. Dr. Wolthusen is a member of the ACM, IEEE, AMS, USNI, and AFCEA.

Naval War College Review, Summer/Autumn 2004, Vol. LVII, No. 3/4

SELF-INFLICTED VULNERABILITIES

Stephen D. Wolthusen

One of the most prominent, if sometimes controversial, figures in software engineering resigned in 1985 from the Panel on Computing in Support of Battle Management of the Strategic Defense Initiative Office, publishing a series of essays declaring it unlikely that the program would meet the goals implicitly set forth by President Ronald Reagan for the SDI program.[1] Two decades later, this assessment has gained in pertinence as transformation technologies become reality and reliance increases on network-centric operations and C4ISR* assets to achieve critical operational objectives. Concern has spread even to the level of individual tactical units, while the potential persists for damage or at least costly friction and lost options at the strategic level.[2]

While information technology has become a highly efficient force multiplier in a large number of roles—from producing transparency in logistics flows to providing target data for strike packages in near real time to guiding munitions themselves—there are differences between information systems and other engineering artifacts that are dangerous to ignore. As information system components suffuse what had previously been the domain of mechanical engineering, as well as similar disciplines, these engineering artifacts frequently come to rely on information technology for their core functionality and hence take on the properties previously associated only with pure information systems.

Software engineering has made only limited progress in producing large, reliable, and trustworthy information systems. Developing such systems (or even their software components) that can be mathematically proven, or at least argued convincingly, to be correct and complete is feasible on a relatively small scale, but it remains, given the consequences of faults, a daunting task at the

* Command, control, communications, computers, intelligence, surveillance, and reconnaissance.

scale contemplated and necessary for network-centric operations. Unlike in mechanical artifacts, uncertainty in such design criteria generally cannot be adequately compensated for by safety margins.[3] Any effort to develop trustworthy, high-assurance systems faces limitations as to what can be subjected to independent verification and validation, let alone mathematical proof with the precision and completeness of requirements and specifications.[4] Success is extremely rare.

BUILDING FORTRESSES ON SAND

Two examples suggest the gulf in scope between the systems for which capabilities and correctness have been proven with mathematical rigor and those actually used in mission-critical tasks. The Ship's Helicopter Operating Limits System, initially deployed with the Royal Navy's Merlin helicopter on Type 23 frigates, was developed to the standard of mathematically provable correctness. A highly specialized and experienced team of scientists and engineers required five years to generate 27 KLoC (thousands of lines of code) of proven and verified software.[5] Moreover, there existed a physical system for the software to control, one that could be modeled precisely, complete with kinematic parameters, and that could therefore be described exactly in a formal specification, from which code could be derived without ambiguity.

A contrasting example would be a COTS (commercial off-the-shelf) general-purpose operating system. One of them, Microsoft Windows 2000, contains more than 10,000 KLoC critical to system security and operational capability; depending on metrics used, Microsoft Windows 2003 contains approximately 50,000 KLoC. None of that code is modeled, specified, or implemented in such a way as to permit even evaluation of the trustworthiness of a component running this operating system, regardless of the characteristics of the layered applications. Interactions between the layered application and the underlying operating system escape, by definition, modeling and specification. Despite advances in computer science and software engineering, it is not at all clear that such large demands are within reach of the methods used for smaller systems even if the resources and time for such an effort are not bounded.

This is in large part due to the fact that the complexity of interactions among software components typically increases significantly faster than the size of the code base, and it does so in a superlinear fashion (i.e., typically as a polynomial in the LoC). While strict hierarchical design methodologies and implementations have long been the subject of research, success in the field has been somewhat limited.[6] Even under optimistic assumptions regarding defect rates, therefore, statistical models predict the presence of several thousand defects for such a COTS product—even with the additional caveat noted above, that unspecified behavior can result in ambiguity as to what constitutes a defect. Though the Microsoft Windows 2000 operating system has been certified as meeting the Common Criteria Evaluation Assurance Level 4 for trustworthiness, critical vulnerabilities are still discovered with some regularity, which is extremely likely to continue for the lifetime of the system.[7]

Even worse, it is not sufficient even to have individual components with proven certain security and assurance characteristics; their combination, such as between systems on a network, can still be insecure.[8] Such problems also arise—at levels of rigor far below formal modeling and proof—from configuration variations and the introduction of new subsystems (attached devices, new programs, or program revisions) within a single computer system. The ultimate result is a staggering combinatorial problem that simply cannot be addressed by mere testing, particularly since by definition the types of defects and cascading failure modes must be assumed to be triggered deliberately by an adversary with precise knowledge of the information system, rather than obeying standard probability distributions.

Despite these well known limitations in trustworthiness, assurance, and manageability, off-the-shelf information technology products—for which safety and reliability requirements are generally relevant only as far as the civilian market will bear the inevitable increases in cost and decreases in otherwise desirable features—are increasingly used at all levels in the U.S. Navy, from planning to engineering systems onboard warships. This introduces a significant number of failure modes that must be considered but are nevertheless frequently ignored with predictable results. A case in point is an engineering network casualty aboard USS *Yorktown* (CG 48) in September 1997 that left the cruiser dead in the water for about two hours and forty-five minutes.[9] Land combat systems do not typically have the same levels of complexity—at least, not yet—but the gap is closing rapidly as new electronics subsystems are added and internetworked, as in the M1A2 main battle tank.

NO WAY BACK

Even if it were not already established acquisitions policy, fiscal considerations would dictate that COTS products, or marginal variations on them, will continue to dominate procurement of large parts of C4ISR assets. That is true as well for critical elements of civilian infrastructure that are increasingly relied upon for mission-critical requirements. Even if procurement of custom solutions were considered, such alternatives would lag considerably behind commercially available systems in terms of functionality.[10]

This reliance on commercially available products has already shown its drawbacks in such areas as electronics for weapons systems, where the cost and feasibility of reengineering are even less attractive than for purchasing systems reliable for the "life of type."[11] For software-based COTS systems, the outlook is even bleaker, for a number of reasons. First, hardware components are traditionally designed to a significantly higher level of quality, not least because errors introduced at the design stage are considerably more expensive to correct than with software-based systems. Hardware already manufactured with defects may need to be destroyed, recalled, and, if the defect is found, replaced.

For their part, most commercially available microprocessors have a sizable number of "errata," documents detailing known problems and, where possible, work-arounds. Such errors have occasionally garnered much public attention, with customers demanding replacement of defective parts. Nonetheless, the incentives for software vendors are somewhat different. For them it is cost-effective to ship a product with possible, suspected, or even known defects and, by and large, correct them only when reported from the field. This practice appears to be accepted by virtually all users of COTS products.[12]

Thus, it is frequently possible simply to replace a microprocessor or other electronic component with a newer, functionally equivalent component as it reaches the end of its service life. In software-based systems, however, not only functionality of obsolete and ill-defined software must be reproduced but, frequently, its defects as well. The behavior of the actual system may well depend on fixes and work-arounds installed in the old equipment.

This situation has led, particularly in the financial services industry, to cases of decades-old financial software running on multiple layers of simulated operating systems and "middleware" components—not unlike Russian *matryushka* nesting dolls. Each of these layers introduces its own defects and uncertainties, limiting overall efficiency and ultimately assurance. As a result, presumably, the reliability of complex software-based systems drops. Options to redress this quandary are quite limited, since frequently when defects and vulnerabilities are discovered the remedies require configuration changes (for both hardware and software, the former often necessitating the latter) beyond the immediate corrective measure.

Any effort to develop trustworthy, high-assurance systems faces limitations. Success is extremely rare.

A second, related problem is the tendency of software systems to make use of the rich functionality available in COTS systems or systems assembled from existing components. The dependencies introduced in commercial systems are less well known than for government in-house, or GOTS, components. This introduces a further web of unknowns. Situations can result where repairing a defect in one component generates cascading side effects, possibly rendering the entire system unusable—even when the components are all from a single vendor. These dependencies produce systems for which the traditional last resort of "life-of-type buys" is simply not feasible, particularly once vulnerabilities become publicly known; for which reengineering—just as with civilian systems—is frequently a euphemism for complete redevelopment; and for which assurance in mission-capability declines precipitously over time as new elements and components are introduced into already underdefined designs.

The alternative of developing and maintaining similarly feature-rich systems with provably high assurance, however, is likely not to be palatable to decision makers except under the most dire requirements, and even then it may not be feasible. An example of such an attempt was the onboard flight control software of the Space Shuttle program. This software, though far less complex than that associated with most COTS-based environments and so, one might have expected, less expensive, has cost in excess of a hundred million dollars to maintain.[13] Indeed, the very concept of mechanized proofs of correctness has been the subject of intense scrutiny.[14]

SAME TOOLS, DIFFERENT OBJECTIVE

Despite dire predictions implied by these considerations, and although a number of highly critical situations have been documented, in remarkably few incidents have malfunctioning information systems led directly to loss of life or similarly grave consequences.[15] The reason may be, however, that systems are being built with adequate safeguards, and the complexity of critical systems is being limited not because of laws, regulations, formal mathematical methods, or similar engineering mandates but because engineers are aware of such warnings as those discussed here about software safety and reliability.[16]

However, there exists a marked difference between adequate provision for failure in the majority of civilian application areas and in defense systems, whether they are to be relied on in harm's way or used in supporting roles. Most civilian systems (with some obvious exceptions such as avionics) can ensure safety by shutting down an information subsystem or components.[17] Such a "fail-stop" mechanism, however, is not likely to be an option in defense-related applications, let alone in those used in combat, unless features for recovery and falling back on manual emergency procedures can be employed.

Traditionally, critical applications without fail-stop options, such as flight-control systems, have relied on multiple redundancy and component-based survivability, as well as fallback. This approach, however, implies significant expense, delay, and a need to codify and validate elaborate operating procedures, clearly beyond what is feasible. For most defense-related information systems, one cannot unambiguously demonstrate the effectiveness of redundancy or fallback, and in any case the required decision loops are likely to be faster than even partial manual fallback mechanisms can achieve.

All this implies, especially given the usual circumstances under which defense systems must operate, that even a system known to be in some degree defective can be acceptable (for instance, a sensor that occasionally reports false measurements) if the alternative would be downing it and jeopardizing a mission. What is critical here is to recognize that such failures can and will occur and must be anticipated at the level of overall mission planning and execution.

Even in combat support missions, such as logistics, where time scales are less compressed than in combat itself, flaws or failures in information systems can be extremely detrimental to overall objectives if inadequate consideration has been given to their possibility. While clerical errors can be made, and have always been, without the aid of electronic information systems, the results can be considerably less amusing than the delivery of snow plows and road salt to Danish troops in Basra at the height of the summer of 2003. Such errors could render entire missions impossible if the rapid and unchecked propagation of their effects causes large volumes of data to become invalid or even merely unreliable. The lack of fallback solutions in case of a severe failure of a logistical system, whether caused by an intrinsic defect or a deliberate attack, can severely affect combat readiness or endanger missions. In the worst case, it might be necessary to open each and every container and crate to locate vital items, then to ascertain the location and needs of each unit requiring them.

While information technology has become a highly efficient force multiplier, there are differences between information systems and other engineered components that are dangerous to ignore.

It is therefore imperative to consider, for each use of information systems, the faults that could be induced and the effects, both primary and secondary, they could have on overall mission objectives. This has to be done however mundane an application seems to be, even for commercial desktop and productivity software. It is precisely the improbable and unanticipated side effect that can cause the most significant disruption, as no contingency plans are likely to exist. Technical countermeasures can be identified and taken, but information assurance rests equally on the organizational factors, along with technical prevention, hardening, and countermeasures. Whether information is delivered electronically or on a scrap of paper is largely irrelevant—if it is accurate, complete, and received in time.

A corollary to this observation is to partition information systems in such a way that individual elements to be employed in network-centric warfare (NCW) are developed to desirable levels of assurance (presumably through testing and other verification and validation measures), and independently, to establish baseline capabilities. The additional capabilities provided by the linking and internetworked operation of such information systems must, given the limitations in providing assurance sketched above, be treated as fundamentally ephemeral.

ASSURANCE AND INFORMATION WARFARE

Information warfare thus far has clearly not lived up to the expectations raised during the 1990s.[18] While it would be clearly imprudent to dismiss IW as yet another ploy to focus resources and funding through overstated threat analyses—the threats identified are very real indeed, if somewhat exaggerated—the concept can be carried farther. Information warfare may or may not be useful in the foreseeable future as an instrument of warfare in the narrow sense of subjugating the will of an adversary to one's own, because of the potential impact on civilian populations (an impact that clearly makes it a potential instrument for terrorists or other entities not bound by the Geneva Conventions Relative to the Protection of Civilian Persons in Time of War). Nonetheless, the role of information in warfare can hardly be overstated and has in fact been understood since antiquity.[19] It is precisely in that respect, however, in which forces relying on network and information-centric systems could expose themselves unknowingly (or worse, having ignored known threats) to new modes of attack.

A design criterion for cryptographic protocols has been proposed in which the authors assume "the presence of a hostile opponent, who can alter messages at will. In effect, our task is to program a computer which gives answers which are subtly and maliciously wrong at the most inconvenient possible moment."[20] In designing information systems to take account of their effects on mission accomplishment, or indeed their effect on the planning and execution of missions themselves, the same assumptions need to be made.

Defects in information systems obviously can cause severe damage and disruption even without intervention of an adversary. It is the hallmark of skilled attackers, however, to identify weaknesses and vulnerabilities in information systems that, taken by themselves, might seem insignificant. Beginning with such small openings, attackers will attempt to escalate the damage potential until their objectives have been reached. Sophistication is not always necessary; frequently vulnerabilities exist that make attacks a rather rote, straightforward, and even automatable matter.

Improving information assurance can therefore be considered a two-pronged undertaking. One part consists of identifying the mission assurance category for each system and component, as laid down by the assurance requirements.[21] The possible system failure modes, both internal and external, can then be identified (using, for instance, fault-tree analysis) and remedies or contingency plans devised.[22]

The other part of ensuring overall information assurance is far more challenging. It requires considering effects on other systems that operate (at least nominally) independently or at higher levels and devising similar mitigation and remediation strategies. One might justifiably argue that the onus is on systems "upstream" of the system under review. This argument is valid, however, only under highly idealized circumstances, since it assumes that a system once examined is permanently frozen with regard to its potential harmful effects on upstream systems. To the contrary, and as noted above, even apparently minor changes can invalidate critical assumptions and introduce new failure modes.

The insidiousness of the problem lies primarily in internetworking effects, which can also have transitive detrimental effects across multiple intermediate systems and components. Consider a network with critical systems built upon a vulnerable COTS base into which a piece of malicious code is inserted—rapid spread throughout a possibly monocultural information system network can cripple vital operational capabilities.[23] In aviation safety, one frequently hears of "long, thin chains" leading to the few documented cases of mechanical failure. Such chains exist in information-assurance failures also, but human and organizational elements must be taken into consideration as well. When information assurance is considered this way, defenses against information warfare attacks become a welcome but implicit side effect of overall information assurance, since there is no need to specify deliberate actions an adversary might take, since any such action must already have been considered pursuant to the most cherished law of engineering—Murphy's.[24]

IMPACTS ON NETWORK-CENTRIC WARFARE

Even in highly asymmetric conflicts, the temptation must be resisted to extract maximum economies of force on the basis of an assumption that technological superiority, particularly in information systems, ensures success. There exists a profound danger that the wrong lessons will be learned, particularly from the successes of ENDURING FREEDOM and IRAQI FREEDOM. If they are, future plans and operations will be built on highly brittle foundations.[25]

Operations, whether at the strategic or tactical level, should not be predicated upon the full nominal capabilities of network-centric organization. Account must be taken at the outset of the considerable spectrum of possible degradation or complete failure of information systems and other elements, regardless of cause, but certainly including enemy information operations. Otherwise, missions could commence under overoptimistic assumptions of forces required or objectives possible; success would then require that the vast majority of information system components operate at peak performance for the entire duration of the mission. Plainly, that cannot be safely assumed.

Planning, then, should allow for contingencies that require humans in the loop at critical junctures to transmit and process mission-critical information when automated systems fail, and missions should be structured accordingly. This is, ultimately, the price one has to pay for using highly complex, internetworked systems of low assurance. It also means, however, that the ability to short-circuit the enemy's decision cycle may be degraded considerably at any given time by system failure. To restore overall decision speed and responsiveness, it may be necessary to shift at certain points, if only temporarily, to a hierarchical, pre-network-centric structure, or vice versa.[26] Planning should identify such junctures in advance and explicitly include capabilities that permit information system components to be used effectively even when operating in isolation or only on a small component of the overall network. Ultimately, however, it is information systems, both civilian and defense, that must change to improve survivability and assurance, as more and more military systems are designed that cannot function at all without information components.

NOTES

1. D. L. Parnas, "Software Aspects of Strategic Defense Systems," *Communications of the Association for Computing Machinery* 28, no. 12 (1985), pp. 1326–35.
2. A. K. Cebrowski, J. J. Garstka, "Network-Centric Warfare: Its Origin and Future," Naval Institute *Proceedings* 124, no. 1 (January 1998), pp. 28–35; T. P. M. Barnett, "The Seven Deadly Sins of Network-Centric Warfare," Naval Institute *Proceedings* 125, no. 1 (January 1999), pp. 36–39; D. S. Alberts, J. J. Garstka, F. P. Stein, *Network Centric Warfare,* 2d ed. (Washington, D.C.: Office of the Assistant Secretary of Defense for Networks and Information Integration, 1999); U.S. Navy Dept., *Naval Transformation Roadmap: Power and Access . . . from the Sea* (Washington, D.C.: U.S. Defense Dept., 2003); and J. C. McGroddy, ed., *Realizing the Potential of C4I: Fundamental Challenges* (Washington, D.C.: National Academy Press, 1999).
3. R. Woodhead, "Spirit Rover Humbled by Classic Programming Error," *ACM Risks Forum* 23, no. 14; P. Regan and S. Hamilton, "NASA's Mission Reliable," *IEEE* [Institute of Electrical and Electronics Engineers] *Computer* 37, no. 1 (January 2004), pp. 59–68.
4. The term "assurance" is used in two ways, distinguished by context. In the context of "information assurance," it refers to the guarantee that an entity obtains required information in such a way that a set of constraints is met. The set of constraints contains but is not limited to timeliness, correctness, confidentiality, integrity, availability, and non-repudiability. In a second, more narrow context, assurance refers to developmental assurance or trustworthiness measures following the standard nomenclature of ISO/IEC 15408 (see note 7).
5. S. King, J. Hammond, R. Chapman, and A. Pryor, "Is Proof More Cost-Effective than Testing?" *IEEE Transactions on Software Engineering* 26, no. 8 (2000), pp. 675–86.
6. P. G. Neumann, R. S. Boyer, R. J. Feiertag, K. N. Levitt, and L. Robinson, *A Provably Secure Operating System: The System, Its Applications, and Proofs*, SRI Computer Science Laboratory Report CSL-116, 2d ed. (Menlo Park, Calif.: SRI, May 1980); L. Robinson and K. N.

Levitt, "Proof Techniques for Hierarchically Structured Systems," in *Data Structuring,* ed. R. T. Yeh, vol. 4 of *Current Trends in Programming Methodology* (Englewood Cliffs, N.J.: Prentice Hall, 1977), pp. 173–96; D. A. Peled, *Software Reliability Methods* (Berlin: Springer-Verlag, 2001).

7. International Organization for Standardization and International Electrotechnical Committee, *Common Criteria for Information Technology Security Evaluation: International Standard 15408,* version 2.1 (Geneva: 1999); and U.S. National Institute of Standards and Technology Information Technology Laboratory and U.S. National Security Agency Information Assurance Directorate, *National Information Assurance Partnership Common Criteria Evaluation and Validation Scheme Validation Report: Microsoft Corporation Windows 2000,* Report CCEVS-VR-02-0025 (Washington, D.C.: 2002). Evaluation technical reports are partially withheld as proprietary information.

8. D. McCullough, "Noninterference and the Composability of Security Properties," *Proceedings of the 1988 IEEE Symposium on Security and Privacy (SOSP '88)* (Oakland, Calif.: 1988).

9. G. Slabodkin, "Smart Ship Inquiry a Go," *Government Computer News,* 31 August 1998; and A. DiGiorgio, "The Smart Ship Is Not the Answer," U.S. Naval Institute *Proceedings* 124, no. 6 (June 1998), pp. 61–64.

10. Undersecretary of Defense (Acquisition, Technology, and Logistics), *The Defense Acquisition Systems,* U.S. Department of Defense Instruction 5000.1 (Washington, D.C.: 2003); and P. Oberndorf and D. Carney, *A Summary of DoD COTS-Related Policies,* Technical Report (Pittsburgh, Penna.: Carnegie Mellon Software Engineering Institute, 1998).

11. S. Morrow, "What Comes after Tomahawk?" U.S. Naval Institute *Proceedings* 129, no. 7 (July 2003), pp. 32–35.

12. R. M. Brady, R. J. Anderson, and R. C. Ball, *Murphy's Law, the Fitness of Evolving Species, and the Limits of Software Reliability,* Technical Report 476 (Cambridge, U.K.: Cambridge University, Computer Laboratory, 1999); and R. Anderson, "Security in Open versus Closed Systems: The Dance of Boltzmann, Coase and Moore," *Proceedings of Open Source Software: Economics, Law and Policy* (Toulouse, Fr.: 2002).

13. N. G. Leveson, ed., *An Assessment of Space Shuttle Flight Software Development Processes* (Washington, D.C.: National Academy Press for the Commission on Engineering and Technical Systems, National Research Council, 1993).

14. D. MacKenzie, *Mechanizing Proof* (Cambridge, Mass.: MIT Press, 2001); J. H. Fetzer, "Program Verification: The Very Idea," *Communications of the Association for Computing Machinery* 31, no. 9 (1988), pp. 1049–63; R. A. DeMillo, R. J. Lipton, and A. J. Perlis, "Social Processes and Proofs of Theorems and Programs," *Communications of the Association for Computing Machinery* 22, no. 5 (1979), pp. 271–80; D. MacKenzie and G. Pottinger, "Mathematics, Technology, and Trust: Formal Verification, Computer Security, and the U.S. Military," *IEEE Annals of the History of Computing* 19, no. 3 (1997), pp. 41–59; G. Pottinger, *Proof Requirements in the Orange Book: Origins, Implementation, and Implications,* Technical Report (Ithaca, N.Y.: Cornell Univ., Mathematical Sciences Institute, 1994).

15. P. G. Neumann, *Computer-Related Risks* (Reading, Mass.: Addison-Wesley, 1994); and N. G. Leveson, *Safeware: System Safety and Computers* (Reading, Mass.: Addison-Wesley, 1995).

16. C. A. R. Hoare, "How Did Software Get So Reliable without Proof?" in *FME '96: Industrial Benefit and Advances in Formal Methods, Third International Symposium of Formal Methods Europe,* ed. M.-C. Gaudel and J. Woodcock, Lecture Notes in Computer Science (Heidelberg, Ger., Springer-Verlag, 1996), pp. 1–17.

17. Another exception is the "supervisory control and data acquisition" system. SCADA systems, which link master and remote units and monitoring stations, are typically used to monitor and control plants or equipment distributed over large geographic areas.

18. R. C. Molander, A. S. Riddile, and P. A. Wilson, eds., *Strategic Information Warfare: A New Face of War,* MR-661-OSD (Santa Monica, Calif.: RAND, 1996); R. E. Neilson, ed., *Sun Tzu and Information Warfare*

(Washington, D.C.: National Defense Univ. Press, 1997); R. Henry and C. E. Peartree, "Military Theory and Information Warfare," *Parameters* 28, no. 3 (August 1998); and Z. Khalilzad, J. P. White, and A. W. Marshall, eds., *Strategic Appraisal: The Changing Role of Information in Warfare,* MR-1016-AF (Santa Monica, Calif.: RAND, 1999).

19. L. T. Greenberg, S. E. Goodman, and K. J. S. Hoo, *Information Warfare and International Law* (Washington, D.C.: National Defense Univ. Press, 1998).

20. R. J. Anderson and R. M. Needham, "Programming Satan's Computer," in *Computer Science Today: Recent Trends and Developments,* ed. J. van Leeuwen, Lecture Notes in Computer Science (Heidelberg, Ger.: Springer-Verlag, 1995).

21. U.S. Defense Dept., *Information Assurance,* U.S. Department of Defense Directive 8500.1 (Washington, D.C.: 2002).

22. M. R. Lyu, ed., *Handbook of Software Reliability Engineering* (New York: McGraw-Hill, 1996); and R. Manian, J. B. Dugan, D. Coppit, and K. J. Sullivan, "Combining Various Solution Techniques for Dynamic Fault Tree Analysis of Computer Systems," *Proceedings of the 3rd IEEE International Symposium on High-Assurance Systems Engineering (HASE '98)* (Washington, D.C.: IEEE Computer Society, 1998).

23. M. Newman, "The Structure and Function of Complex Networks," *SIAM* [Society for Industrial and Applied Mathematics] *Review* 45 (2003), pp. 167–256; A.-L. Barabási and R. Albert, "Emergence of Scaling in Random Networks," *Science* 286 (1999), pp. 509–12; D. S. Callaway, M. E. J. Newman, S. H. Strogatz, and D. J. Watts, "Network Robustness and Fragility: Percolation on Random Graphs," *Physical Review Letters* 85 (2000), pp. 5468–71; and R. Albert, H. Jeong, and A. Barabási, "Error and Attack Tolerance of Complex Networks," *Nature* 406 (2000), pp. 378–82.

24. T. Bear, "Murphy's Law Was Born Here," *Desert Wings,* 3 March 1978, p. 3.

25. M. Boot, "The New American Way of War," *Foreign Affairs* 82, no. 4 (July/August 2003), pp. 41–58.

26. C. S. Gray, *Modern Strategy* (Oxford, U.K.: Oxford Univ. Press, 1999); and G. T. Hammond, *The Mind of War: John Boyd and American Security* (Washington, D.C.: Smithsonian Institution Press, 2001).