Research on Techniques and Tools for Computer Security: The COAST Project and Laboratory

Director: *Eugene H. Spafford*
Department of Computer Sciences
1398 Computer Science Building
Purdue University
West Lafayette, IN 47907–1398
spaf@cs.purdue.edu
+1 317 494-7825 (office)
+1 317 494-0739 (fax)

Last revision: 30 September 1992

Abstract

The goal of the COAST project is to establish a long-term research program to investigate new approaches for computer security and computer system management. The focus will be on techniques and tools for common, non-trusted (in the DoD, "orange-book" sense) systems. The project includes establishment of a laboratory for this research, and the development of educational materials. We intend for the research to be broad-based and draw from experience and expertise in other areas of computing such as software engineering, language design, operating systems, computer networking and databases.

COAST will be supported through grants and donations. Our sponsors will get early access to COAST products, both software and publications. COAST sponsors will get special access to the faculty and students conducting the research and will help direct the research they conduct. We expect continuing, direct feedback from our sponsors on our research directions and results. We also expect that our research results will have a significant impact on the current state of practice as our results are brought into use.

Introduction

It is clear that computer security is an area of increasing, major concern and that the computing community is facing some severe challenges related to security. Incidents related to disclosure of information, wide-scale computer breakins, and the exponential growth in the number of computer viruses being written and discovered all indicate an increasing threat to effective use of computing resources.[1]

The recent report *Computers at Risk* [29], forcefully outlines several critical security problems facing computer users. Our increasing reliance on computers for critical applications poses increasing temptation for unauthorized criminal and terroristic activity. Our increased connectivity provided by new network technologies simply amplifies the threat. Additionally, the increased use of computer databases, and the

[1] See, for instance, [28].

increasing use of parallel and distributed processing, all add new variables that have not, as yet, been adequately examined. Initiatives that link together computing systems from around the world and that provide access to more users, such as the National Research and Engineering Network (NREN) and High Performance Computing (HPC) initiatives, will likely add to the potential for security problems.

Computer security, as an area of specialization, is difficult to accurately define. Even professionals working in the area of computer security have difficulty agreeing on an exact definition of the field. Part of the reason that security is difficult to define is because it draws heavily upon so many areas of computing. In at least one sense, it seems closely related to software engineering — computer security is devoted to ensuring that software and hardware meet their specifications and requirements when used in a potentially hostile environment. Computer security thus includes issues in computer system specification, verification, testing, validation, safety, and reliability. However, security encompasses much more than these issues, including topics in (at the least) operating systems design, architectural design, information security, risk analysis and prediction, database organization, encryption and coding, formal models of computation, fault tolerance, supportive interface design, government regulation and policy, managerial decisions, security awareness, and education.

The difficulty in defining computing security is also reflected in the scattered and underdeveloped educational and research programs in the area. While other fields of computing research have well-defined bodies of literature, the field of computer security is largely defined by chapters in textbooks on operating systems, data communications, and databases. The field currently has only a few widely-circulated archival journals with refereed articles on computer security topics: *Computers & Security*, *Cryptologia*, and the new *Journal of Computer Security*. Few universities or colleges offer in-depth education in computer security. When courses are taught, relatively few textbooks on computer security are in use, and several of the most commonly used ones are principally devoted to cryptography (e.g., [8]). Research in academia is being done by a limited number of faculty at scattered locations working with a few students; senior faculty at many institutions do not view security as a "serious" area for research and specialization. What research is being done, in academia or commercially, is primarily oriented to military needs.

This lack of visibility, training, and coordinated research efforts has led to a shortage of practitioners trained in *practical* computing security. It has also contributed to a lack of consideration of security issues when new computer systems are being designed, thus placing those new systems at risk. As technology propels us into a future where global networks of communicating, multi-vendor computer systems are commonplace, the lack of universally-accepted social norms and laws will lead to difficulties that only well-designed computer security tools and techniques may prevent.

Research in computer security-related issues has usually been conducted under a number of different rubrics reflecting its cross-disciplinary nature. Work in areas such as computer architecture, operating systems, data communications, database systems, and software engineering has included questions of computer security. Despite advances in all these areas, most direct security-related research in the last few decades has been largely directed towards only a few selected areas. Most of the research done to date has been in support of formal trust models for multi-level secure machines employed in military settings, including compartmented-mode workstations. The results of this research is usually of little use in "real-world" computing environments.

In particular, research in computer security methods and protocols over the last few decades has largely been focused on issues of theoretical models of secure systems, intrusion detection systems, and communications security issues (e.g., cryptography). Little research has been focused on the development of tools for improving general security, audit techniques, availability models, countering malicious software (e.g., computer viruses and worms), policy formation, and integrity methods. Perhaps this is because work in all these areas requires an interdisciplinary approach. It may also be because the traditional focus of research has primarily been in military systems computer security, which focuses on issues of confidentiality [15, 16]. Thus, there has been less support for research in the area of designing security tools and techniques for everyday use on commercial and educational computing platforms than on military systems.

Research Goals

Computer security is an area of concern that has recently achieved a sense of urgency. Most security tools for computer systems are either proprietary, complex security systems, or else they are tools that are capable of being a threat as well as a defense: many of these security tools can be misused and employed as system cracking tools, and others are easy to subvert to hide malicious activity. For instance, password checker programs may be used to help ensure that users have selected "strong" passwords. However, these same programs may be used to "crack" the passwords of a target system. Likewise, tools to check for file permissions or the presence of bugs may be used to probe for a system's weak spots.

The COAST (**C**omputer **O**perations, **A**udit, and **S**ecurity **T**ools) Project is directed to computer security research on more general and practical tools and techniques. The focus of this research is on typical computing environments — systems without multi-level requirements, and without formal levels of "trust." In particular, the research is directed to developing approaches to increasing the security of existing systems without severely impacting their usability. Our goal is to explore how to increase confidence in existing systems in a cost-effective and user-friendly manner.

At Purdue over the last five years, I have been exploring issues in *practical* computer security. My work has included widely-cited work in analysis of malicious code such as viruses (e.g., [22, 23, 26, 27]). In 1991, I coauthored the award-winning book *Practical* UNIX *Security* [11], now considered the standard reference in the field. I have also been involved in work on static audit and analysis tools. An initial result of this work was the COPS [10] security audit tool for UNIX systems, used worldwide on tens of thousands of computer systems. This tool runs on over a dozen varieties of UNIX, and detects several score configuration and management problems that may lead to security problems. It too is viewed as the standard of its type.

Another tool, OPUS, [25, 24] is currently being designed and experimented with as one of the first COAST projects. This tool will help prevent users from choosing weak or obvious passwords, yet it will not add to the administrative burdens of the system administrator, nor will it require the megabytes of disk storage that would be required of a comprehensive dictionary-based screening mechanism of similar power. Furthermore, an explicit design goal was to prevent the tool from being useful as a means of breaking passwords. Experience has shown that over 80% of attacks from the outside of networked systems are based on exploitation of weak passwords[2], so we expect OPUS to also enjoy widespread use.

Research is also being conducted on a portable integrity checking tool to search for unauthorized changes to files, as might be made by the addition of a trapdoor, logic bomb or virus (*Tripwire*, currently in beta test). We are also developing an efficient scanner tool that can be configured to search for indicated bit patterns (e.g., a virus signature); this reconfigurable virus-scanning tool will be described at a forth-coming conference [14].

[2]According to unpublished reports from the DARPA CERT/CC and other incident response centers.

A graduate student supported under a NASA Graduate Research Fellowship is working with me on scheduling support for fully autonomous distributed systems [4]. This research is devoted to development of mechanisms and protocols that support cooperation of various of machines in a network without forcing them to give up local control of resources or scheduling autonomy. A direct result of this research is support for distributed scheduling that cannot result in denial of service because each individual machine may control allocation of its own internal resources at all times.

Under examination for possible future development within COAST are a COPS-like tool for auditing network security and configuration; a network "immune system" that can respond automatically to threats introduced across a network; new methods of testing software to discover potential security vulnerabilities; methods of secure distribution of vulnerability information; development of mechanisms to counter denial of service attacks; and developing a classification of audit information to separate anomoly detection, accounting, and record-keeping functions.

Overall, the goal of COAST is to conduct coordinated, cooperative research in the area of security tools and methods such as all of the ones described above. A primary focus of this work is to investigate methods of developing and employing practical security tools that are of little or no threat to systems. This research into tools is thus directed towards development of simple, reconfigurable tools that do not require secrecy or extreme privilege to be effective, and that cannot be easily used for system penetration, nor be easily altered to introduce "trojan horses." We also intend to develop formalisms for the development of change-resistant (e.g., virus-proof) software, as well as the development of generic but effective tools and techniques. This work should also result in new insights into the development of secure software.

COAST research will focus on practical application of the techniques developed, with experimental trials in existing systems, at Purdue and elsewhere, to validate the methods used. We hope that sponsors will participate in these trials, to provide us with the kind of real-world constraints and environments that are so seldom present in the experiments conducted by other researchers.

Another explicit goal of the COAST project is to provide students with the opportunity to be involved with computer security issues. By providing research and educational opportunities, we hope encourage good students to specialize in the area. We also hope to use these results to develop new curricular materials in computer security. By encouraging participation of COAST sponsors, we further intend to widen the base and reach of the computer security educational materials that we expect to produce. The result should be students better equipped to undertake a career in industry or academe with computer security as their specialty.

Other projects — both research and educational — may be undertaken at the suggestion of COAST sponsors, based on the availability of personnel and resources to conduct that research.

Sponsors

Companies and government agencies may become sponsors of the COAST project and laboratory. Sponsors of the laboratory will help determine the direction of the laboratory, through regular feedback, and through an advisory panel. Sponsors will receive early releases of prototype software and our reports. We also anticipate other interactions with sponsors, including placing students for internships, personnel exchanges, and tutorials. The nature of the research proposed for COAST is such that active collaboration of sponsors will be sought to validate and refine COAST products.

Sponsors will be prominently recognized in COAST publications and products (unless they wish to keep a lower profile). Sponsors will be given early disclosure of products, and will have preferential access to faculty, staff and students involved in COAST projects. COAST sponsors will have opportunity to provide significant input into the direction of COAST research. Other, special support will be provided to sponsors as the project matures.

We are interested in joining forces with additional sponsors for the lab. We are seeking a steady, multi-year commitment of support from each sponsor rather than one-time, large grants; most of the projects planned for COAST will be multi-year in scope. Single donations for specific projects and purposes are still appropriate, however, and constitute sponsorship.

As part of the Department of Computer Sciences' support for the COAST project, any sponsor providing at least $40,000 of support for COAST will automatically become a supporting industrial affiliate of the Department during each calendar year of support. This provides additional benefit in the way of access to research being conducted in the Department as a whole, access to faculty and students, and participation in selected events. (Contact Professor Spafford for more details.)

Companies and agencies may also become contributors to (instead of sponsors of) the COAST project. This will entail a smaller donation or gift to the project, and will not result in the same levels of support or disclosure as for sponsors. However, we will seek to involve and support COAST contributors more closely than non-contributors.

Sponsorship of COAST may be in one of four ways:

1. **By unrestricted grant**. A sponsor may make an unrestricted grant of money to Purdue University, on behalf of the COAST project and the principal investigator. This may then be used to support worthy students, fund publication and software distribution, send COAST personnel to appropriate conferences, and purchase equipment and software for the lab. This is the prefered form of support: grant money is subject to a much lower overhead charge by Purdue than other forms of support. Grants and gifts of money and/or equipment may have certain Federal and state tax advantages for the donor, too.

2. **By directed contract**. A sponsor may specifically fund one or more particular projects within COAST, or fund the project as a whole. This involves negotiating a contract with the Purdue Research Foundation for specific deliverables (e.g., reports or software), and would need to include coverage of salaries, equipment, travel, supplies, secretarial expenses, and overhead costs.

   Because of the basic philosophy behind COAST, and because of the university mandate for publication of research results, offers of contracts for proprietary research are not solicited. Likewise, COAST will not accept funding that would result in research results being subject to a government review prior to release.

3. **By fellowship**. A sponsor may wish to directly offer multi-year, continuing fellowship support to graduate students (or undergraduates) working on COAST-related projects. These fellowships may be named and designated largely as the sponsor wishes. These would be funded as awards to the principal investigator, and designated specifically as fellowships for students involved in COAST research. Sponsorships involving an on-site internship are possible.

4. **By equipment and software donation**. One of the goals of the project is to develop and maintain a state-of-the-art laboratory for computer security research and education. This requires an *on-going*

procurement of equipment and software. Sponsors may donate *appropriate* equipment and materials to help stock and maintain this lab. Any equipment so donated will largely define the "baseline" systems used for development and distribution of COAST products.

Prospective sponsors are invited to contact Gene Spafford for further information about opportunities to sponsor COAST projects and participate in the project and its results.

Personnel

Principal Investigator

Gene Spafford attended the School of Information and Computer Sciences at Georgia Institute of Technology, holding both a Georgia Tech President's Fellowship and a National Science Foundation Graduate Fellowship. He received the Ph.D. in 1986 for his design and implementation of the Clouds distributed operating system kernel, and for his contributions as one of the original members of the Clouds design team. Next, Dr. Spafford spent a year and a half as a research scientist with the Software Engineering Center at Georgia Tech. His duties there included serving as a principal software engineer with the Mothra software testing project.

In 1987, Professor Spafford joined the academic faculty of the Department of Computer Sciences at Purdue University. At Purdue, he has taught courses in operating systems, software engineering, networking and data communications, and issues of ethics and professional responsibility. He is also an active member of the Software Engineering Research Center (SERC) there. At Purdue, Spaf has conducted research on methods of increasing the reliability of computer systems, and the consequences of computer failures. This has included work with computer security, software testing and debugging, and issues of liability and professional ethics.

Professor Spafford is the Associate Editor of the journal *Computing Systems*, and is on the advisory and editorial boards of the *International Journal of Computer and Software Engineering*, the *Virus Bulletin*, *ICSA Virus News and Reports*, and the *Journal of Information Systems Security*. He has written extensively in the field of computer security, including coauthoring an award-winning book on UNIX Security ([11]) and a widely-cited book on computer viruses ([26]). He has also spoken internationally at panels, conferences, symposia, and colloquia on these issues.

Dr. Spafford is a member of the Association for Computing Machinery (ACM) and serves as chairman of that organization's Self-Assessment Committee and of its ISEF Awards Committee, as well as being a charter member of the new Technical Standards Committee. He is a Senior Member of the IEEE and of the Computer Society of the IEEE. He is also a member of both the Usenix Association, and of the Computer Security Institute. He is an associate member of the National Research Center on Computing and Society and is a founder of the Purdue Computer Emergency Response Team (a FIRST member organization[3]). In 1992, Spaf was elected to serve a two-year term on the international FIRST steering committee.

[3]FIRST is the Forum of Incident Response and Security Teams.

Other Personnel

The Department of Computer Sciences at Purdue University is the nation's oldest computer science department, with a long tradition of research and service in computer systems, software engineering and computer security. It is usually ranked as one of the top dozen programs in computer science in the country. The department includes dedicated labs for several major projects and centers, including the Computing About Physical Objects (CAPO) project and the Software Engineering Research Center (SERC).

The department graduates over a dozen Ph.D. candidates in computer science each year from an outstanding graduate student population of almost 200. The current faculty of over three dozen researchers includes internationally-known individuals with interests that include networking, computer systems, databases, algorithms and data structures, user interfaces, graphics and visualization, parallel computation, computational theory, software engineering, performance evaluation, artificial intelligence, and robotics.

Several Purdue faculty members are also interested in computer security topics. What follows are brief descriptions of a few of the faculty who have expressed strong interest in collaborating on projects as part of the COAST effort. Their interests and areas of expertise help provide additional depth an potential to the COAST effort.

Bharat Bhargava

Professor Bhargava received a Ph.D. in electrical engineering from Purdue University in 1974. He is a full professor at Purdue with an international reputation in the areas of distributed databases and distributed concurrency control. He is also known for his work in fault tolerant transaction processing. Professor Bhargava has served in numerous professional offices, including as a member of the editorial board of *IEEE Transactions on Knowledge and Data Engineering* and as Vice-Chair of the IEEE Technical Committee on Distributed Processing.

Dr. Bhargava is director of the RAID project at Purdue — research in replication control, checkpointing, and communications in distributed database systems [2, 1]. The RAID project includes experimental work on a distributed database system developed in his lab. Other experiments are also directed towards dynamic reconfiguration of the system under various conditions, including failure.

Several topics in distributed systems security are of interest to Dr. Bhargava. One in particular is the possible relationship between fault tolerance and access protection. In practice, the differences appear to be small between an access failure caused by a processing fault and caused by an access violation. This suggests that there may be some approaches to the implementation of distributed access control that uses already-understood mechanisms from distributed concurrency control. This is something that might be easily examined within the context of the RAID system. It may also lead to new insights into providing fault tolerance for trusted systems.

Douglas E. Comer

Professor Comer received his Ph.D. from The Pennsylvania State University in 1976, and is a full professor at Purdue. He has exhibited a broad range of interests, primarily in computing systems. He is internationally reknowned for his experimental research and design work in operating systems and network protocols.

Doug is perhaps best known for his best-selling books on the Xinu [5] operating system, and for his series of books on the TCP/IP protocol suite [6, 7]. He is also director of several research projects and labs at Purdue, including the Xinu, MultiSwitch and Cypress projects. He is a former chair of the DARPA Distributed Systems Architecture Board and of the CSNet Technical Committee. Among other offices, he is currently the editor of the journals *Software Practice & Experience* and *Journal of Internetworking — Research and Practice.*

Dr. Comer is interested in many aspects of security, especially as they apply to networks and network communications. One area of particular interest that would be a perfect fit with the goal of COAST is to explore methods of identifying malicious code in network traffic and alerting system managers before the code is able to damage systems. This becomes a very challenging problem in networks with gigabit-per-second network traffic flow. The goal of this work is to develop strong network security mechanisms without loss of speed or capacity.

Prasun Dewan

Professor Dewan received the Ph.D. in computer sciences from the University of Wisconsin at Madison in 1986. He is an assistant professor at Purdue with research interests in structure editing, user interface generation, collaborative systems and distributed systems. He is the principal investigator of the SUITE project, funded by NSF and the SERC, involving research in a distributed, object-oriented collaborative system.

Prasun is currently directing some research in security issues that arise in collaborative environments. Specifically, he is looking at an easy-to-use and flexible access control model for such environments [19]. The access model is based on a general model of collaboration developed as part of the work on the SUITE collaborative system [9]. His research involves identifying the various dimensions of sharing in a collaborative environment and associating an access right with each of these dimensions. It divides subjects, protected objects, and access rights into hierarchical groups and allows users to associate access rights with both these groups and the individual members of a group. A multidimensional inheritance model allows a more specific access definition to override a general one.

Future research will involve examining the policies that might drive these mechanisms, and defining rules to resolve conflicting definitions. Research will also be devoted to the design of data structures and algorithms to efficiently search the inheritance space.

Edgar Knapp

Professor Knapp received his Ph.D. from The University of Texas at Austin in 1992. His dissertation work was concerned with the design of correct and reliable concurrent programs using stepwise refinement. He joined the Department of Computer Sciences at Purdue University in Fall 1992 as an assistant professor.

Edgar is interested in all aspects of distributed system design and implementation. In particular, one of the prime concerns in the development of large software systems is their complexity. Modularization and layering are important software engineering techniques that may be used to manage this complexity. Module and layer boundaries are often chosen such that a separation of distinct concerns is achieved. Each of these concerns may then be addressed separately. Security issues represent concerns that can and should be singled out in this manner.

With the approach of splitting a system into a set of interacting parts comes the problem of integrating these parts to form a working whole. Dr. Knapp has developed a language and tools based on formalisms developed by Chandy and Misra. He has also developed a method of refinement to design large software systems in a systematic and scientific manner, and has applied his work to a variety of practical problems (c.f. [12, 13]). It seems desirable to address security concerns in a modular fashion, and this line of inquiry is an ideal candidate for further research and evaluation.

John T. Korb

Dr. Korb received his Ph.D. from the University of Arizona in 1979, where he participated in the design and implementation of the Icon programming language. He has held both academic and research positions with the Computer Sciences faculty at Purdue in the past dozen years. He is currently the Director of Research Facilities for the department. His research interests include operating systems, programming languages, networks, and user interfaces. He has lectured widely on the X Window System, and is involved with many of the research projects at Purdue.

Working with the departmental computing facilities, Dr. Korb's responsibilities make him acutely aware of the current problems with practical computer system security, both from technical and administrative points of view. He is keenly interested in performance monitoring and testing tools to detect and correct security problems, including intruders and viruses. Additionally, he is interested in the development of new techniques and tools for managing a multi-vendor, networked installation: currently, he is responsible for almost 300 machines running over a dozen different operating systems.

Vincent F. Russo

Professor Russo received his Ph.D. in computer science from the University of Illinois at Urbana-Champaign in 1990. His dissertation work was in object-oriented operating systems; he designed the *Choices* operating system there. Since joining the faculty at Purdue as an assistant professor, Vince has been actively involved in research in the areas of operating system design and object-oriented systems.

Currently, Vince is directing research on the *Renaissance* project [17]. One goal of this project is to provide the ability for unrelated applications written in a variety of programming languages to obtain access to objects distributed throughout a network of machines and implemented in a variety of manners [18]. The approach being used is to create a first-class description of an object's interface that has a runtime realization and is independent of any programming language or system. A translator is then provided to convert this description into a variety of languages and systems.

An example of this would be providing object-oriented access to files resident on an NFS file server. These descriptions will be used to generate interfaces to remote objects dynamically at runtime. They will also allow flexible but complete interface specification providing both increased encapsulation and documentation. The intent is to make it possible to map non-object-oriented substrates into systems by viewing objects as servers and using the server/client model common to many non-object-oriented distributed applications.

A central issue in any distributed object model, including this one, is security: deciding who has access to an object and with what permissions. Dr. Russo and one of his graduate students are investigating how to provide this security in an efficient and safe manner. He is also interested in using the *Renaissance* system as an operating system testbed for further experiments in security enhancements to operating systems.

Samuel S. Wagstaff, Jr.

Professor Wagstaff obtained his Ph.D. in mathematics at Cornell University in 1970. He has worked in Computational Number Theory ever since then. For the past dozen years his research has focused on Public Key Cryptography, especially on the computational complexity of factoring integers. While at the University of Georgia in 1981–83, he helped to design and build a special processor for factoring large integers. In 1983, he joined the faculty of the Department of Computer Sciences at Purdue University. He has taught a course in Cryptography and Data Security almost every year since then, as well as teaching courses in several other areas.

Dr. Wagstaff is an Associate Editor of the journal *Advances in the Theory of Computation and Computational Mathematics*. He is coauthor of a popular book on factorization of numbers of the form $b^n \pm 1$ [3]. He has given dozens of lectures around the world on Computational Number Theory, and has published extensively in this and related areas (e.g., [20, 21]).

Sam's interests include most areas of computer security. He hopes to work with the COAST Project on matters related to cryptography. His interests mesh especially well with COAST projects involving authorization, applying zero-knowledge proofs techniques, and integrity checking methods.

Michal Young

Professor Young received the Ph.D. in computer sciences from the University of California, Irvine, in 1989 and joined Purdue as an assistant professor. In 1991 he was honored by selection as an NSF Presidential Young Investigator.[4]

Michal's research interests focus on improving the dependability of concurrent (distributed, parallel, and multi-threaded) software. He is currently directing research on modular state-space analysis of concurrent software systems (using process algebra to avoid combinatorial explosion when analyzing the interactions among many processes [30] and concurrency control and scheduling for hard real-time systems. He is also pursuing research in software development environments and integration of testing, analysis, and verification techniques [31].

One of Professor Young's recent research projects involves generation of state-space analysis tools for combinations of (possibly domain-specific) descriptions of processes. A typical use of generated tools would be detection of races, deadlocks, and other coordination problems in an evolving or heterogeneous system. Another potential use is in analyzing vulnerabilities by modeling a system design together with attacks of various kinds. This could be used to discover and analyze security weaknesses in real-time and highly concurrent computing systems.

An area of interest for future COAST research, shared with Professor Spafford, is the modeling of availabilty of computational resources and developing methods to ensure continued availability. Denial of service is a significant problem in security, and research in this area is sorely needed.

[4] Support of COAST research involving Professor Young may qualify for matching funding from the National Science Foundation.

For More Information

A current version of this document, as well as many of the cited documents, is available for anonymous ftp from the host `ftp.cs.purdue.edu` in the directory `pub/spaf/security`. Professor Spafford may also be contacted directly for further information about COAST.

References

[1] B. Bhargava, E. Mafla, and J. Riedl. Communications in the RAID distributed database system. *Computer Networks and ISDN Journal*, 1991.

[2] B. Bhargava and J. Riedl. RAID distributed database system. *IEEE Transactions on Software Engineering*, pages 726–736, 1989.

[3] John Brillhart, D. H. Lehmer, J. L. Selfridge, Bryant Tuckerman, and S. S. Wagstaff, Jr. *Factorizations of* $b^n \pm 1$*, b = 2, 3, 5, 6, 7, 10, 11, 12 up to high powers*, volume 22 of *Contemporary Mathematics*. American Mathematical Society, Providence, RI, second edition, 1988.

[4] Steve Chapin and Eugene H. Spafford. Scheduling support for an internetwork of heterogeneous, autonomous processors. Technical Report TR–CSD–92–006, Department of Computer Sciences, Purdue University, West Lafayette, IN, 1992.

[5] Douglas E. Comer. *Operating Systems Design: The Xinu Approach*. Prentice Hall, 1984.

[6] Douglas E. Comer. *Internetworking with TCP/IP: Principles, Protocols, and Architecture*. Prentice Hall, 1991. 2nd edition.

[7] Douglas E. Comer and David L. Stevens. *Internetworking with TCP/IP: Design, Implementation, and Internals*. Prentice Hall, 1991.

[8] Dorothy E. R. Denning. *Cryptography and Data Security*. Addison-Wesley, Reading, MA, 1983.

[9] Prasun Dewan and Rajiv Choudhary. A high-level and flexible framework for implementing multi-user interfaces. *Transactions on Information Systems*, 1992.

[10] Daniel Farmer and Eugene H. Spafford. The COPS security checker system. In *Proceedings of the Summer Usenix Conference*. Usenix Association, June 1990.

[11] Simson Garfinkel and Gene Spafford. *Practical Unix Security*. O'Reilly & Associates, Inc., Sebastapol, CA, 1991.

[12] Edgar Knapp. An exercise in the formal development of parallel programs: Maximum flows in graphs. *ACM Transactions on Programming Languages and Systems*, 12(2), 1990.

[13] Edgar Knapp. Derivation of parallel programs: Two examples. *Science of Computer Programming*, 1992. To appear.

[14] Sandeep Kumar and Eugene H. Spafford. A generic virus scanner in c++. In *Proceedings of the Eighth Computer Security Applications Conference*, December 1992.

[15] National Computer Security Center. Trusted computer system evaluation criteria. Technical Report DoD 5200.28-STD, U.S. Department of Defense, 1985.

[16] National Computer Security Center. Computer security subsystem interpretation of trusted computer system evaluation criteria. Technical Report NCSC-TG-009, U.S. Department of Defense, 1988.

[17] Vincent Russo. Process scheduling in the Renaissance object-oriented multiprocessor operating system. In *Proceedings of SEDMS II*. Usenix, 1991.

[18] Vincent F. Russo and Patrick A. Muckelbauer. Distributed object interoperability via a network type system. In *Proceedings of the Workshop on Object Orientation in Operating Systems*. IEEE, 1991.

[19] Honghai Shen and Prasun Dewan. Access control for collaborative environments. In *Proceedings of the Conference on Computer Supported Cooperative Work*. ACM, 1992.

[20] J. W. Smith and S. S. Wagstaff, Jr. How to crack an RSA cryptosystem. *Congressus Numerantium*, 40:367–373, 1983.

[21] J. W. Smith and S. S. Wagstaff, Jr. Methods of factoring large integers. In D. V. Chudnovsky, G. V. Chudnovsky, H. Cohn, and M. B. Nathanson, editors, *Number Theory, New York, 1984-85*, volume 1240 of *Lecture Notes in Mathematics*, pages 225–234. Springer-Verlag, 1987.

[22] Eugene H. Spafford. The Internet Worm: Crisis and aftermath. *Communications of the ACM*, 32(6):678–687, June 1986.

[23] Eugene H. Spafford. An analysis of the internet worm. In C. Ghezzi and J. A. McDermid, editors, *Proceedings of the 2nd European Software Engineering Conference*, pages 446–468. Springer-Verlag, September 1989.

[24] Eugene H. Spafford. Observations on reusable password choices. In *Proceedings of the 3rd Security Symposium*. Usenix, September 1992.

[25] Eugene H. Spafford. Opus: Preventing weak password choices. *Computers & Security*, 11(3):273–278, May 1992.

[26] Eugene H. Spafford, Kathleen A. Heaphy, and David J. Ferbrache. *Computer Viruses: Dealing with Electronic Vandalism and Programmed Threats*. ADAPSO, Arlington, VA, 1989.

[27] Eugene H. Spafford and Stephen A. Weeber. Software forensics: Can we track code to its authors? In *Proceedings of the 15th National Computer Security Conference*, October 1992.

[28] Spectrum, August 1992. Special issue on data security.

[29] National Research Council System Security Study Committee. *Computers at Risk: Safe Computing in the Information Age*. National Academy Press, 1991.

[30] Wei Jen Yeh and Michal Young. Compositional reachability analysis using process algebra. In *Proceedings of the 4th Symposium on Testing, Analysis and Verification*, pages 49–59. ACM SIGSOFT, October 1991.

[31] Michal Young and Richard N. Taylor. Rethinking the taxonomy of fault detection techniques. In *Proceedings of the 11th International Conference on Software Engineering*, pages 53–62, May 1989.