Journal of Information Warfare

Volume 2, Issue 3

Contents

Editorial – *W. Hutchinson* iii

Conferences iv

Authors v

Information Security Education Theme Papers

Teaching Competitive Intelligence and Cyberwarfare in a Business Context 1
H.L. Armstrong, J. Davey

Teaching Hands-On Network Security: Testbeds and Live Exercises 8
G. Vigna

Certification and Accreditation: A Program for Practitioner Education 25
K.L. Burke, C.W. Rasmussen, C.E. Irvine, G.W. Dinolt, T.E. Levin

Security Dimension of IT in Developing Countries: Risks and Challenges 38
R. Casmir, L. Yngstrom

Enhancing Information Warfare Education Through the Use of Virtual and Isolated Networks 47
D.J. Ragsdale, S.D. Lathrop, R.C. Dodge

Design and Implementation of an Information Security Laboratory 60
V. Padman, N. Memon, P. Frankl, G. Naumovich

Teaching Coursework Master – Computer Forensics 72
C.J. Armstrong

A Tutoring System for IT Security Education 79
M. Schmitt, J. Hu, Ch. Meinel

Using Outcomes-based Assessment as an Assurance Tool for Assurance Education 86
S. Older, S Chin

General Papers

If You Go Down to the Internet Today – Deceptive Honeypots 101
S. Yek, C. Valli

Accessing Security Incident Information on the Internet 109
M.A. Belsis, A.N. Godwin, L. Smalov

Is the IW Paradigm Outdated? A Discussion of U.S. IW Theory
T.L. Thomas 117

Journal of Information Warfare
© Copyright 2003

Published by:
School of Computer and Information Science,
Edith Cowan University, Western Australia
ISSN 1445-3312

Online version published by:
Teamlink Australia Pty Ltd
ISSN 1445 3347

www.Jinfowar.com

Editorial

This issue is slightly different than normal. The papers have been selected from those submitted to the Third World Information Security Education Conference held at the Naval Postgraduate School, Monterey, California in late June, 2003. The papers were chosen by the conference organisers – Cynthia Irvine (Naval Postgraduate School, Monterey, California) and Helen Armstrong (Curtin University, Western Australia), and were re-written for the journal. The theme for this edition of JIW is Information Security Education.

Also two papers that should have been included in the last issue are in this edition, one by Yek and Valli, and another by Belsis *et al*.

The opinion paper is by Thomas on US Information Warfare theory.

Finally, I would like to welcome Professor Daniel Kuehl of the National Defense University, Washington, onto our esteemed Advisory Board.

I would also like to thank all the authors for their contribution and encourage you all to keep on submitting your papers.

William Hutchinson

August, 2003

Conferences

InfoWarCon 2003, Washington. Sept 30 to Oct 3, 2003
http://www.infowar.com

3rd Australian Information Warfare and Security Conference, Adelaide
Nov 20-21, 2003.
No site on publication date. Email: jill.slay@unisa.edu.au

1st Australian Information Security Management Conference, Perth,
Western Australia. Nov 24, 2003.
http://www.we-bcentre.com

Authors

Colin J Armstrong is an associate lecturer with the Curtin University School of Information Systems, Western Australia, where he focuses on teaching. His research interests of information security, information warfare and computer forensics.

Helen Armstrong is a Senior Lecturer within the School of Information systems at Curtin University, Perth Western Australia. Helen's work experience includes accounting, programming, systems analysis, project management, IS management, IS auditing, IT consulting and university lecturing. Her interests in the area of security include computer crime, network security, Internet security, information warfare, biometrics and security management. Helen's doctoral studies related to the security of medical informatics. Helen is the international chair of the IFIP WG11.8 on Information Security Education.

Meletis Belsis has been a member of the DKERG research group for the last three years. His main research interests are concentrated on Information Security. Additional interests include internet technologies and database development. Meletis completed his Honours degree at Coventry University in the area of Computer Science in 1998. He continued for a two year Masters by Research degree in Information Systems. After finishing his Masters by Research, he started a PhD degree in the area of Information Security at Coventry University. The main area of interest in his current research is security modelling and the development of an Enterprise Security Model.

Karen L. Burke is a Research Associate Professor in the Department of Computer Science at the Naval Postgraduate School. She has 16+ years as an information system security engineer with the US Air Force and in private industry. She has developed Information Assurance policies and guidance, has performed accreditations, and has been a team member on the development of multilevel secure systems. She is currently constructing courses and lectures in certification and accreditation. She is a member of the Armed Forces Communications and Electronics Association.

Respickius Casmir is a PhD student at the Department of Computer and Systems Sciences (DSV). He started doing research in Information and Communication Systems Security field in 2001. He recently successfully defended his degree of Licentiate of Philosophy with specialization in IT Security Education. Respickius is also an employee of the University of Dar es Salaam in Tanzania. His other area of specialization is IT Security Management. Before switching to IT security career, he was working as a network engineer.

Shiu-Kai Chin is a Professor in the Department of Electrical Engineering and Computer Science at Syracuse University and the Program Director of Computer Engineering. He is Director of the New York State Center of Advanced Technology in Computer Applications and Software Engineering (CASE). His research applies mathematical logic to the engineering of highly assured systems. He has worked closely with the Information Warfare Branch at the Information Directorate of the Air Force Research Lab in Rome, NY.

Jack Davey is the Assistant Director, Defence Security Authority for the Australian Department of Defence in Canberra, Australia. Prior to joining the Department of Defence, Jack worked in electronic engineering and communications in the aviation industry. One of Jack's current areas of responsibility is security education within defence and the training of cyberwarriors.

George W. Dinolt is an Associate Professor of Computer Science at the Naval Postgraduate School. Dr. Dinolt has 20+ years as a researcher and developer in the area of computer security in academia and in industry. His principal research focus is mathematical models of security. He has been a security architect on several projects and was part of a team that developed a very high assurance system for the US Department of Defense, where he also developed the mathematical models of security for the system.

Major Ronald C. Dodge, Jr. is an Aviation officer in the US Army and is a member of the Army Acquisition Corps. His current assignment is Assistant Professor and Deputy Director of the Information Technology and Operations Center (ITOC) at the U.S. Military Academy. Throughout his service a the Military Academy he has been a forceful proponent of student-centered learning. His innovative research and teaching in Information Assurance and Cybersecurity have brought him national recognition.

Phyllis G. Frankl is a Professor of Computer Science at Polytechnic University in Brooklyn, New York, USA and the Director of the Wireless Internet Center for Advanced Technology (WICAT). She received a B.A. degree in mathematics and physics from Brandeis University, an M.A. degree in mathematics from Columbia University, and M.S. and Ph.D. degrees in computer science from New York University's Courant Institute of Mathematical Sciences. Prof. Frankl's research interests include software engineering, wireless information systems and computer security.

Ji Hu received his MS in computer science from the Beijing Polytechnic University. Since October 2001, he is a doctoral candidate at the Institute for Telematics/ Department of Computer Science, University of Trier, Germany. His research interests are information security and security education.

Cynthia E. Irvine is an Associate Professor of Computer Science at the Naval Postgraduate School and Director of the Center for Information System Security Studies and Research (CISR). She has spent the past 15 years developing theoretical foundations, design, implementation, and analysis of high assurance secure systems. She has developed security aware applications for high assurance systems and contributed to the development of a Class A1 security kernel. She has authored over 70 papers in the area of trusted computing. Dr. Irvine is a senior member of the International of Electrical and Electronics Engineers, a member of the Association for Computing Machinery and serves on the Senior Advisory Board of the Association of Old Crows.

Major Scott Lathrop was commissioned as an Armor officer in 1990 and is now serving as a Systems Automation officer in the US Army. His current assignment is Assistant Professor and Senior Research Scientist in the Information Technology and Operations Center (ITOC) at the United States Military Academy. Major Lathrop was the primary architect for the IWAR/VIAN Laboratory during the period in which this lab gained international notoriety. Without his dedicated efforts the VIAN concept, as it is described in this article, would not have been realized. Major Lathrop's research interests include: information assurance and cyberwarfare.

Timothy E. Levin is an Associate Research Professor at the Naval Postgraduate School. He has spent over 17 years working in all aspects of secure computer systems research and development, including the design of security features and formal verification for A1 operating systems and relational database management systems. His current research interests include management and quantification of security in heterogeneous networks, data integrity in distributed systems, and the application of formal methods to secure computer systems. Mr. Levin is a member of the Association for Computing Machinery and International of Electrical and Electronics Engineers.

Christoph Meinel studied Mathematics and Computer Science at the Humboldt-University in Berlin from 1974 to 1979. He received his PhD degree in 1981. From 1981 to 1991, he worked at the Department of Mathematics at the Humboldt University and at the Institute of Mathematics of the Academy of Sciences in Berlin as a scientific assistant. In 1988, he received his habilitation degree. In 1992, he became a full professor for computer science at the University of Trier. He is visiting professor at the Luxembourg International Advanced Studies in Information Technology as well as the graduate school of the computer sciences department of the Beijing University of Technology, China.

Nasir Memon is an Associate Professor in the Computer Science Department at Polytechnic University, New York. He received his B.E. in Chemical Engineering and M.Sc. in Mathematics from the Birla Institute of Technology, Pilani, India and received his M.S. and Ph.D. degrees from the University of Nebraska, in Computer Science. He has held an assistant professor position at Arkansas State University and at Northern Illinois University. He was also a visiting faculty at Hewlett Packard Laboratories, Palo Alto, California in 1997-98. His research interests include Data Compression, Computer and Network Security, Multimedia Data Security and Multimedia Communications.

Gleb Naumovich received a PhD in Computer Science from the University of Massachusetts, Amherst in 1999. His research interests are in the areas of software engineering, programming languages, application security, and intellectual property protection. His current work focuses on optimizing model checking tools for software. He also works on developing static techniques for detection of security bugs in distributed Java programs.

Susan Older is an Associate Professor in the Department of Electrical Engineering and Computer Science at Syracuse University, as well as the Education Director for the Systems Assurance Institute. Her research interests include the semantics of programming languages, concurrency theory, logics of programs, formal methods, and their collective use to support reasoning about complex system behavior.

Vikram Padman is a graduate student at Polytechnic University in Brooklyn, NY. He received his B.S. degree in Computer Engineering from Polytechnic University in 2002. His current research focuses on developing efficient host intrusion detection systems.

Lt. Colonel Ragsdale, Ph.D., has served for over twenty-two years as an officer in the US Army. During this time he served in a variety of important operational, and research and development assignments. Currently he is an Associate Professor and Director of the Information Technology and Operations Center (ITOC) at the US Military Academy. His current research focuses on information security, Information Assurance (IA), and Information Warfare. He is a frequent speaker and panelist at national and international IA conferences and he has published dozens of papers and articles on IA topics.

Craig W. Rasmussen is an Associate Professor in the Department of Applied Mathematics at the Naval Postgraduate School, where he has been since 1991. For most of the intervening years, his principal research interest has been graph theory. He has more recently developed a keen interest in applications of mathematics to information assurance and secure communications. He is a member of the Society for Industrial and Applied Mathematics, the American Mathematical Society, and the Mathematical Association of America.

Michael Schmitt studied computer science with focus on computational linguistics at the University of Koblenz, Germany. From 1996 to 2001, he was a research assistant at the Institute for Telematics at the Medical University of Lübeck, Germany. In October 2001, he joined the Institute for Telematics in Trier. In April 2003, he passed his doctoral examination on automatic test generation based on formal specifications at the University of Göttingen. His current research activities focus on information security.

Yek Suen is a research student within the School of Computer and Information Science at Edith Cowan University, Perth Western Australia. Suen holds a Bachelor of Science and a Bachelor of Business, and is currently undertaking research incorporating deceptive strategies used in wireless honeypots.

Craig Valli is a computer security lecturer and researcher in the School of Computer and Information Science at Edith Cowan University. His research interests are in the areas of network security, defensive network deception, network intrusion detection, firewall, SOHO security and beowulf computer clustering. Craig consults to industry on network management and network security issues and holds several industry level networking certifications.

Giovanni Vigna is an Assistant Professor in the Department of Computer Science at the University of California in Santa Barbara. His current research interests include network and computer security, intrusion detection, security of mobile code systems, penetration testing, and distributed systems. In particular, in the last years he worked on STAT, a framework for the modular development of intrusion detection systems. He also published a book on Security and Mobile Agents and he is the Program Chair of the International Symposium on Recent Advances in Intrusion Detection (RAID 2003). Giovanni Vigna received his M.S. with honors and Ph.D. from Politecnico di Milano, Italy, in 1994 and 1998, respectively.

Louise Yngström has been engaged in research and education in IT security with her department since 1985. She is a pioneer in interdisciplinary holistic IT security education, Research interests, apart from security curricula and education, are security issues related to society and organisations. Previous research areas were IR- systems and computer aided instruction. She is a member of several IFIP (International Federation for Information Processing) working groups related to her interests and the Swedish national representative to IFIP TC 11. In her department, she is the director of SecLab and the director of PhD studies.

Teaching Competitive Intelligence and Cyberwarfare in a Business Context

H.L Armstrong[1], and J. Davey[2]

[1] School of Information Systems
Curtin University, Australia,
E-mail: H.Armstrong@curtin.edu.au

[2] John Davey
Department of Defense, Canberra
Research Fellow, Edith Cowan University, Australia,
E-mail: jack.davey@defence.gov.au

Abstract

Competitive intelligence has risen in profile over the past decade to become a key contributor to the survival of organizations operating in highly competitive environments. It is seen as an essential management tool in the current global marketplace. Several authors claim there is a need for the inclusion of competitive intelligence in Masters level studies however the area is rarely included in educational programs offered by tertiary institutions. This paper describes a course in competitive intelligence and cyberwarfare offered within IT-related Masters programs at Curtin University. The dominant teaching tool used is scenario work simulating realistic organisational situations incorporating competitive intelligence and cyberwarfare activities.

Introduction

The need for more education and training in the areas of cyberwarfare and competitive intelligence has emerged from a variety of sources. A number of factors have contributed to this need including the rise in crimes utilizing computers and the use of the Internet by organized criminal and terrorist groups. The increasing reliance and utilization of global communications networks by business organizations for survival and dominance over competitors is also a contributing factor.

This paper looks at the use of competitive intelligence by today's business organization and its use in the modern executive's information warfare toolkit. The need for inclusion of education in the area of competitive intelligence and cyberwarfare in tertiary postgraduate studies is discussed together with an overview of a unit of study in this area currently taught in masters degrees within the School of Information Systems at Curtin University.

Competitive Edge versus Annihilation

McCarthy (2003) reports that business organizations in the US suffered losses estimated at $100 to $250 billion (USD) in sales due to industrial espionage in 2000. A survey in May 2001 of US manufacturing companies reported 55% admitted spying on their rivals (Vacca, 2002). There is a fine line between competitive intelligence

ISSN 1445-3312 print/ISSN 1445-3347 online

gathering and industrial espionage, and many of the tools used in industrial espionage are also harnessed in offensive and defensive cyberwarfare.

Competitive intelligence is undertaken in many different forms in both public and private organizations. Miller (2000) suggests that government agencies conduct intelligence focused more on threats than on opportunities, but in corporations this situation is reversed, with emphasis more on opportunities than threats. The line separating ethical and unethical activities in the gathering of information on rivals and the means employed to gain a competitive advantage appears to be perforated and it is easy to slip between the holes. Shake and Gembicki (1999) suggest modern business executives are equipped with tools of combat with well-appointed fortresses where information warfare in a business context involves achieving and maintaining an information advantage over competitors.

Competitive intelligence is seen as an essential part of the modern organization just as Sun-Tsu considered it an essential part of warfare strategy in 400BC. It incorporates both intelligence (analyzing gathered data about rivals) and counterintelligence (protecting ones own information sources). Intelligence is not confined to the military domain, and Kanaher (1998) suggests it is imperative to corporate organizations due to the rapid pace of business, information overload, increased global competition from new competitors, more aggressive competition, rapid technological change and forceful global changes in international trade agreements. The race to survive in a cut-throat global marketplace is on. Jones recommends every morning you ask 'what can I do to beat Company Z today' as neither your competition nor technology will wait for you (Jones et al., 2002).

The heightened awareness of competitive intelligence has been spirited by increased global competitiveness characterized by increased industry consolidation and fragmentation (Fleisher & Blenkhorn, 2001). The Internet provides both the data and tools for competitive intelligence, offering a wealth of information and search bots for those wishing to gather information about corporations and individuals. The Internet also provides a wealth of tools that can be used to immobilize and threaten a rival's survival. In order to adequately protect against such an attack, one must be aware of not only methods and tools of attack but also one's own vulnerabilities and likely strategies and responses of adversaries.

Although competitive intelligence is not a recent phenomenon in the business community and awareness of the benefits of competitive intelligence to organizations is evident, the tertiary education industry has been chided for lack of response to this need. Fleisher and Blenkhorn (2001) state that competitive intelligence is rarely included in MBA programs, and Shaker and Gembicki (1999) believe competitive intelligence is an essential ingredient to effective management as a manager's knowledge is derived from both formal and information education. They go on to suggest that the IT culture at large and education programs, at the Masters level in particular, neglect the area of competitive intelligence.

Teaching Competitive Intelligence and Cyberwarfare

The unit of study presented at Curtin University is titled Business Intelligence and Cyberwarfare and is primarily a unit covering intelligence, competitive intelligence and corporate information warfare strategy formulation and response. The unit is

primarily undertaken by students enrolled in the Masters of Internet Security Management, but can also be taken by students enrolled in Masters programs in Information Technology, Information Systems, and Electronic Commerce. Students require pre-requisite knowledge in Internet security and network architecture in order to enrol in the business intelligence and cyberwarfare unit.

The overall objective of the unit relating to competitive intelligence is to develop in students an understanding of the nature of competitive intelligence and its purpose in contemporary business environments. The main objectives of the cyberwarfare section is to develop in students an understanding of cyberwarfare in defence, business and government organizations, the need for security of information and systems within public and private organizations, to build a knowledge of the strategies and tools used in cyberwarfare and to help students identify appropriate security mechanisms to protect against industrial espionage and cyberwarfare.

As the unit encompasses both competitive intelligence and cyberwarfare the topics covered include the competitive business environment, asset evaluation and identification, information systems and networking, intelligence and counterintelligence, information collection and analysis methods, counterintelligence by deception and denial of service, conflict in cyberspace, information systems protective security, insider system attacks, external system attacks, intrusion detection, incident reporting and situational awareness, reaction to attack, damage control and business continuity, national infrastructure issues and practical use of hacking and intrusion detection software tools.

The unit is conducted as an intensive course over two consecutive weekends. The content is practical in nature with the classes conducted in laboratories and small tutorial rooms. The main laboratory used is a sandpit lab which can be connected to the Internet or operate as an isolated peer-to-peer network. The ability for the lab to be isolated is important when running software such as packet sniffers, network mapping tools, spyware and root kits. Connecting to the Internet presents the risk of attack from outside sources while students are becoming familiar with the software tools.

In 400BC Sun Tsu advised ‘be so subtle that you are invisible, be so mysterious that you are intangible; then you will control your rival’s fate’. The unit is taught by staff from both business and defence backgrounds and perhaps it is the prior experience in submarine warfare systems that makes Sun Tsu’s words so appealing in this educational context. The unit is designed to provide knowledge in both strategy and practical tools. The laboratory sessions illustrate the network tools used to both gather information and provide offensive and defensive cyberwarfare capability and the scenarios and group activities teach strategy, tactics and management.

The Use of Scenarios

Several scenarios are used in the unit to enhance learning and place the theory in context in a simulated situation. Students are given a warm-up exercise, an actual case of sabotage in industry, to investigate. The students work in pairs and are required to answer a number of questions relating to the given incident. The final part of the exercise is to take on the mindset of the saboteur and identify ways the perpetrator could have caused more damage. The student’s ideas are then shared with

the class. In summing up salient points of the exercise, a number of questions are asked of the group, including "What does your partner now know about the way you think?", "How much information are you now willing to share with your partner?", and "How well could your partner or the rest of the group predict your action or reaction to a given situation based upon the knowledge they now have about the way you think?". The main objectives of this exercise are not only to use some tools for gathering intelligence, but also to raise students' awareness of the confidential nature of information and the way information can be used.

One of the most effective tools used in the teaching of this unit is a scenario carried out over two of the four days. This scenario features two organizations competing for a government contract to build a specific product (for example, the rockets to launch communications satellites or a specialized biometric authentication device). Each organization is to build a prototype and the government will decide the successful product. Immense government funding has been made available to both organizations for the development of their prototypes. The two organizations are in a highly competitive situation and in order to survive, need to develop a leading edge over their rival. This requires gathering information regarding the activities of the other, turning it into intelligence, and devising strategies to gain a leading edge. Although the details of the industry and products change each time the unit is run, the basic premise of the exercise is the same. Details of the scenario used in the initial running of the unit can be found in Armstrong and Davey (2003).

The overall aim of running the scenario is to allow students to apply methods of intelligence gathering, design competitive intelligence and counterintelligence strategies and learn from the human interaction. Specific aims require students to devise strategies for gathering information about their rival and determine not only the short and long-term impact of their actions, but also the ethical considerations. They must also devise appropriate counterintelligence measures to ensure their competitor does not gather information about them. The scenario exercise provides an opportunity for students to apply some of the theory and tools previously read in books and articles or covered in lectures and labs. In addition, the exercise gives the students some insight into the complexity of the application of these tools and methods, not only with relation to the technology but also the human element, all within an organizational setting.

Running the Scenarios

The scenario was run for the first time in the second half of 2002 and subsequently in the first half of 2003. The scenario is carried out progressively over the two weekends, allowing the students to make notes, consider strategies and develop ideas away from the classroom. A maximum enrolment of 26 students is allowed based upon laboratory capacity. The students are from a variety of backgrounds and nationalities, with different knowledge and expertise, thus adding to the richness of interactions.

Students are assigned roles within the two organizations mainly covering executive functions and lower level technical roles. The scenario becomes more complex as the situation unfolds. An election results in a change of government and the opposition party now in power is not as enthusiastic about the project as the previous

government. A cut in funding of the project is rumoured, then announced, shortly after the change of government.

Information is fed to the executives of the business organizations as the scenario progresses, requiring decisions and actions. Information on government decisions and funding is leaked and the media release press reports also containing confidential information. Updated financial data is given to the financial executives in each organization, requiring management to review staffing levels and outgoings related to the research and development. This information necessitates a review of current organizational operations and tactics within the broader competitive situation in order to not only survive but also maintain a competitive edge.

To make the scenario more realistic informants are planted in each organization as spies and as parties to intimate personal relationships with members of the opposing firm. Early in the exercise the chief executive of each organization is advised of their spy's identity in the adversary's camp but the personal relationships are not divulged until the end. It is up to the executives to maximise opportunities to meet with their informants and collect information.

In both the semesters this unit has been run the groups have contained dominant students who regularly try to coerce other members of the group, or take control and implement actions without due consideration to consequences. As the exercise progressed the proposed actions became more radical and unethical, and the reserved members of the group gradually became more outspoken in their opposition to these suggestions. The intelligence and counterintelligence strategies developed by the groups contained not only the usual intelligence gathering techniques, but also some radical suggestions bordering on the unethical. Some examples included causing physical harm to key employees of the rival organization, locking up the employees and their families, infiltrating the rival organization (executives, cleaning staff, maintenance technicians, etc), hacking into the opposition's computer networks and not only copying information but also modifying engineering formulae and counterfeiting data in order to sabotage the product development. Students are encouraged to consider the implications of their suggestions before any action is taken, bearing in mind that any action will invite a reaction.

It was observed in both semesters that the focus of the groups in forming counterintelligence strategies was purely based upon external threats with little or no consideration of threats from within their own organization. Very few, if any, students were cognizant of the spy in their own ranks until the end of the exercise. Recommended counterintelligence measures did not include means of detecting or monitoring internal threats. As a result proprietary information was passed on to the rival organization unchecked.

The feedback from the students in both semesters has been very positive, with all students stating they learned significantly from the scenario, particularly regarding intelligence, counterintelligence, strategy, ethics, security and many lessons in human relationships.

The lab sessions were well received, and students enjoyed using the software tools. However it is important to integrate the content of the lab sessions with the strategies

and theories to provide a more holistic perspective. This allows the students to build the bigger picture and more easily meet the learning objectives of the unit. Many of the students requested the inclusion of a full cyberwar exercise in the laboratory, however this is not a primary activity required to meet the learning objectives and the limited duration of the course does not allow the time for a full offensive-defensive exercise. The unit is designed to cover only stages one and two of the five-staged training approach suggested by Davey and Armstrong (2001), although some of the tactics and strategies included in stage five of this model are included in the scenarios but not carried out in the physical network environment.

Conclusion

The scenarios and practical exercises aided in achieving the learning objectives for the course and students found these to be of great academic value. Students also valued and learned from the group interaction. Not only were they able to devise strategies and partially apply them, but were also made aware of the complexities and constraints of human interactions within a dynamic business environment.

Although other universities and military academies provide more in-depth practical application of cyberwarfare tools and conduct cyberwarfare games, the course described in this paper is a business strategy unit, conducted within the faculty of the Curtin Business School. The focus of the unit is strategy first, then tactics and tools to support that strategy. It is important to realize that the technology is only part of the entire picture and the course aims at a coverage broader than an in-depth investigation of the hardware and software tools. Dearth (2001) emphasizes that we must think through the conceptual challenges before rushing out to put tactics, techniques and procedures in place. He believes 'cerebral flexibility' will be the hallmark of successful decision-makers and commanders in the future. In this course on competitive intelligence and cyberwarfare we have tried to generate a spark for later ignition.

References

Armstrong, H. & Davey, J. (2003) Assembling Competitive Intelligence using Classroom Scenarios, in Irvine, C. & Armstrong, H. (Eds) *Security Education and Critical Infrastructures*, IFIP / Kluwer Academic Publishers, Boston, pp 159-168

Davey, J. & Armstrong, H., (2001) An Approach to Teaching Cyber Warfare Tools and Techniques, *Journal of Information Warfare*, Vol 1, Issue 2, December, pp 87-94

Dearth, D.H., (2001) Implications and Challenges of Applied information Operations, *Journal of Information Warfare*, Vol 1, Issue 1, September, pp7-15

Fleisher, C.S. & Blenkhorn, D.L., (2001) *Managing Frontiers in Competitive Intelligence*, Quorum Books, Connecticut, USA

Jones, A., Kovacich, G.L. & Luzwick, P.G., (2002) *Global Information Warfare: How Businesses, Governments and Others Achieve Objectives and Attain Competitive Advantages*, Auerbach Publications, Washington DC, USA

Kahaner, L., (1998) *Competitive Intelligence*, Touchstone Books, New York, USA

McCarthy, L., (2003) *IT Security: Risking the Corporation*, Prentice-Hall, New Jersey, USA

Miller, J.P., (2000) *Millennium Intelligence: Understanding and Conducting Competitive Intelligence in the Digital Age*, Cyberage Books, New Jersey, USA

Shaker, S.M. and Gembicki, M.P., (1999) *The WarRoom Guide to Competitive Intelligence*, McGraw-Hill, New York, USA

Vacca, J.R., (2002) *Computer Forensics: Computer Crime Scene Investigation*, Charles River Media Inc., Massachusetts, USA

Teaching Hands-On Network Security: Testbeds and Live Exercises

G. Vigna

Department of Computer Science
University of California, Santa Barbara
E-mail: vigna@cs.ucsb.edu

Abstract

Teaching practical network security requires the use of tools and techniques to support the educational process and to evaluate the students' newly achieved skills. Two fundamental tools that support a hands-on approach to network security are testbed networks and live exercises. Testbed networks provide a safe environment where the students can experiment with the techniques and security tools that they learn about. Live exercises represent a valuable tool to test the students' newly acquired skills and to teach the students the dynamics of network-based attack and defense techniques. However, testbed networks and live exercises are difficult to set up and to manage. For this reason, there are very few courses that use dedicated network testbeds and/or offer live exercise as an integral part of the class work. This paper describes a series of testbed networks and live exercises that have been used in a graduate-level Computer Science course on network security and intrusion detection. Each testbed network is described in detail and its pros and cons discussed. Then, for each live exercise, the setup, execution, and lessons learned are discussed. The intended audience of this paper is represented by instructors – especially in colleges and universities – who want to start using this type of instructional tools but have no experience and are unsure of the possible pitfalls in their design and implementation.

Introduction

Computer security has become a critical issue that affects our everyday life. For this reason, most colleges and other educational institutions have devoted a consistent amount of resources to develop courses and curricula that involve security training (Bishop 1997, Bishop 2000). Typical courses include cryptography, general computer security, network security, and specialized topics, such as firewalls, security of wireless networks, etc.

Most of these courses are taught using standard educational tools, such as textbooks, slides, and papers. In addition, assignment and tests are mostly paper-based and theoretical.

In few instances, courses are characterized by a practical, hands-on approach. These courses are seldom offered because of the additional difficulties of teaching practical security. First of all, security permeates a wide range of technologies. Addressing a comprehensive set of practical security techniques without getting lost in the details of each technology requires careful selection of the topics and particular attention to the way the topics are presented to the students. Second, the instructor is faced with the

1445-3312 print/ISSN 1445-3347 online

difficult decision of choosing the right balance between a completely theoretical approach and a completely practical one (Bishop 1999). Usually, this choice is heavily influenced by the educational curriculum. For example, if there is an existing course that covers the foundational principles of security (e.g., the Bell-LaPadula model), then it is possible to use the course as a prerequisite and focus on more technical issues. Third, teaching practical security requires substantial effort on the part of both the instructor and the educational institution within which the course is given. This makes it difficult to organize practical security courses, particularly in higher education public institutions, like universities, where resources are scarce.

In addition, if the class has a high-impact technical content (e.g., the class covers break-in techniques), then there is a general concern that the class may get 'out of hand' or that a particular institution may be flagged as a 'hacker school' (Brandt 2003).

Nonetheless, we advocate that teaching practical security is important because of the critical nature of the topic. A security education curriculum that covers only the theoretical aspects of security and does not give the student the opportunity to experiment in practice with security technologies cannot prepare the students to the complexity and difficulties of doing research and development in the computer security field.

This paper describes the author's experience in teaching a graduate-level course on 'Network Security and Intrusion Detection'. This course has been taught three times in the past two years. This class differs from traditional courses in security in three ways: a strong practical and technical emphasis, the support for hands-on experience through the use of testbed networks, and the use of live exercises. These three aspects are discussed in the following three sections.

Teaching Practical Security

The goal of the course is to describe in detail the techniques used to violate the security of computer systems and the mechanisms and tools used to both prevent and detect the attacks. The course follows a hands-on approach that gives the students the skills necessary to actually reproduce the attacks and develop the defense tools. For example, the lack of boundary checks in software is not just covered from a theoretical, general viewpoint: Buffer overflow attacks are examined in detail, showing exactly how to create the necessary conditions for an attack to be successful and the tools needed to build and deliver the attack. In addition, the tools used to prevent and detect these attacks are analyzed in details, highlighting the strengths and limitations of each solution.

The rationale behind this approach can be summarized by the well-known saying: 'The Devil is in the details.' That is, only by understanding the low-level details of vulnerabilities and attacks it is possible to avoid the introduction of similar flaws in software and to design protection and detection mechanisms that are actually effective.

A possible argument against this approach is that students are taught 'How to break in'. Some instructors (and some administrators) fear that by teaching how attacks are

actually designed and delivered will create a new generation of hackers that will participate in illicit activities using the skills gained in the security courses.

The obvious counter-argument is that locksmiths know how to break into a house or a car. Locksmiths are not considered criminals and the best locksmiths are those who design the safes where we store our most valuable items. In addition, even though the information about how to violate the security of a system was once in the hands of a few knowledgeable and skilled programmers, nowadays the same knowledge is accessible to the public through the Internet.

Bruce Schneier comments on a recent controversy about teaching viruses are shared by many security experts:

'[…] If we have any hope of improving computer security, we need to teach computer security. Teaching computer security includes teaching how attacks work. It includes teaching how viruses work. It includes teaching how worms work. The bad guys have all sorts of resources to learn how to write viruses. SQL Slammer source code has been available on the Internet. Neither of these two actions will help the bad guys. But they probably will help the good guys. Worms, viruses, exploits, hacking code...they're not infectious diseases. We need to look at them as educational tools, and not things to keep secret.' (Schneier 2003)

The key here is an ethical approach to security. It is the responsibility of the instructor to sensitize the students to the ethical aspects of security and to inform them of the possible consequences of their actions. Therefore, a security class must always include a detailed, in-depth discussion of computer ethics, network etiquette, policies that regulate the department where the course is taught, and computer crime laws.

Hands-on Experience and the root Problem

A hands-on, practical course on network security and intrusion detection cannot be taught with just a textbook, a blackboard, and a few slides. When teaching practical security it is necessary to allow the students to experiment with the security techniques covered in class. The main problem is that most security experiments require privileged access to the operating system – in UNIX lingo, root privileges.

Departments very seldom provide instructional labs where students (even graduate students) have privileged access and can experiment with security tools and techniques. This is mainly because of the possible dangers associated with such activities and because of the complexity that arises in managing an infrastructure with these characteristics. This situation was also the status at the Computer Science Department of Santa Barbara, where no such educational tool was available.

For the first edition of my class on network security, I was able to obtain the permission of the department to create an instructional network testbed where students were allowed to experiment with various types of attacks and defense techniques without disrupting the normal educational activity. Unfortunately, the department resources were limited and, therefore, it was not possible to take away from existing educational laboratories the hardware and software needed for a testbed network. However, the department at that time had gone through a major renovation of its computer network. Therefore, I was able to build a testbed network using pieces of

hardware that were discarded during the upgrade. The resulting testbed was composed of ten hosts (PCs and Sun Workstations) configured with a number of different operating systems. The testbed (named 'the playground' by the students) was accessible remotely and secured by a firewall, so that only authorized users could use it. This network proved to be an invaluable educational tool: For the first time the students were able to test security tools and attacks in a 'safe' environment. The feedback from the students in the class was overwhelmingly positive, and for both the second and third instances of the class the department provided support to improve the testbed network.

Even though the creation, configuration, and maintenance of the testbed required a substantial amount of additional effort from both the instructor and the teaching assistants, this experience proved that it is actually possible to create an instructional tool where students are able to enjoy a hands-on experience with security techniques.

Evaluating the Students with Live Exercises

Even though the testbed network was an important instructional tool, it didn't provide a realistic experience of the attack/defense process. Each tool and technique was experimented with in an isolated way. In addition, there was the need for evaluating the skills that the students acquired during the class.

Therefore, in the first edition of the course I decided to create a live exercise that would give the students a feel for the difficulties of both attacking and protecting computer networks in real time. During this exercise, which was conducted at the end of the class, the students were divided into two teams. The teams had to perform a coordinated attack and defense process against each other, within a limited time frame (around four hours). The students had to provide a report of their activities that allowed the instructor to evaluate their ability to put to work the techniques learned in class. The enthusiastic response of the students convinced me to include this type of exercise in every future edition of the course.

Road-map

Network testbeds and live exercises are invaluable educational tools for a course on practical security. However, they also are incredibly difficult to setup and manage and there are many lessons that were learned from conducting these activities. For each edition of the class, the lessons learned suggested improvements to the structure of the testbed network as well as modifications in the organization and execution of the live exercises. This paper describes in detail the testbed networks used during the course and the structure and execution of each live exercise, explaining the rationale behind them and discussing the lessons learned. The intended audience for this paper is higher-education instructors that may want to use testbed networks as an educational tool and reproduce similar exercises.

The rest of this paper describes the testbed networks and the live exercises used in the first three editions of the class. In the next section, a first simple testbed network is described. Then, a classic Red Team/Blue Team exercise that was run on that network is presented. The following two sections present an improved testbed design and the evolution of the first exercise into a 'capture the flag' contest. Then, two sections detail an improved testbed and a network-based 'treasure hunt', which were used in the third edition of the class. Then a section contains the discussion of some related

coursework. Finally, the last section draws some conclusions and outlines future work.

A First Testbed

The network testbed used in the first edition of the class was very simple. The topology of the testbed was flat, with a single Ethernet hub connecting a number of hosts. At that time, it was decided to use Linux RedHat 6.2 and Windows 2000 for the PCs and Solaris 2.5 for the SparcStations. A whole class C IP subnetwork (254 possible hosts) was dedicated to the testbed. It was particularly difficult to obtain from the department administrators a range of routable IP addresses, given the scarcity of IP addresses that almost all Universities are experiencing.

The network was separated from the Department network by a firewall. This allowed for careful filtering of traffic from and to the outside. In particular, the testbed network was accessible only from three instructional labs used by the students. Traffic to and from the Internet was blocked.

The students were asked to choose testbed account names that were different from the ones used in the Department network. This was done to avoid confusion between the activity performed on the testbed and the normal use of the department network. The students were given the password of the root account on the testbed machines (with the exception of the firewall).

In order to monitor the usage of the network the students were requested to send an email message to the instructor before starting to use the testbed and after having used it. The format of the email message was defined precisely, and contained the time at which the 'session' started/finished, the username and testbed handle of the user, and the IP address of the host the user was connecting from. For example, user Mark Twain (user name `mtwain` with testbed handle `bzero`) connecting to the testbed on January 31st, 2001, at 8:32am from a host with IP address 128.111.48.69 would have sent a message to the instructor with the following subject:

```
TESTBED USER bzero BEGIN 0101310832 mtwain 128.111.48.69
```

A similar message would have been sent when the session was terminated. The use of these messages allowed the instructor to devise scripts that assessed the testbed overall usage and, to a certain degree, to verify that the connections from the outside were associated with legitimate use of the testbed network. Note that this accounting mechanism was not bullet-proof and was created mainly to gather data about the students' usage of the testbed.

During the course of the class, several lessons were learned. First of all, even though the flat topology of the network is easy to build and manage, it does not allow one to compartmentalize different parts of the network. This proved to be a problem during the preparation of the Red Team/Blue Team exercise (see next section), when the class was divided in two competing teams. In some cases, the activity of one team interfered with the productivity of the other team and this sometimes prevented the students from using the testbed to its full extent.

A second important lesson was learned when particularly 'destructive' attacks were tried by the students. Even though the students were told to be very careful in controlling the execution of attacks, in some cases the attacks permanently modified the operating system of the testbed hosts, making the attacked host unusable. In these cases, the operating system had to be reinstalled from the original distribution. This activity was particularly time consuming.

Another unexpected problem was the use of the department network passwords within the testbed. If a testbed user opened a connection from a testbed host to a host in the department network and the login procedure required a password, then the password could have been captured by a number of means. Therefore, the students had to be made aware of the problem of typing passwords in any environment where the keystrokes could be logged by a number of means including kernel-level logging.

A Red Team/Blue Team Exercise

This exercise was carried out during the first edition of the course. In this exercise, the class was divided into two teams: the Red Team and the Blue Team. The Red Team was responsible for attacking and compromising a set of hosts, while the Blue Team was responsible for detecting the attacks and, in a limited form, for protecting the hosts.

The final goal of the Red Team was to obtain a file named `secret.txt` stored on each victim host. There could be multiple copies of the file and decoy copies could also be present. The only files that had to be retrieved were those whose contents started with the keyword `SECRET`.

The goal of the Blue Team was to detect the attacks coming from the Red Team. In addition, the Blue Team could execute some counter-measures to slow down or confuse the attackers. In particular, the Blue Team could freely decide where to store the secret file. The only requirement was that the file be on a mounted file system.

Some rules were introduced in order to make the exercise more interesting. First of all, the Blue Team could not filter or block any network traffic. Second, the Blue Team could not patch any vulnerability: The Blue Team had to work with out-of-the-box operating systems. These rules were imposed to prevent the Blue Team from completely patching and locking down the systems. Even though in real-life situations network access to sensitive services is actually heavily filtered, in this case a network filter and the patching of known vulnerabilities would have made the whole exercise uninteresting.

The Read Team also had some limitations. First, the Red Team could not use *a priori* knowledge about the victim hosts. It was clear that some of the hosts in the class testbed would have been used as victims. The students were invited to avoid any use of 'testbed-specific' knowledge, e.g., the association of a certain Ethernet address with a certain host in the testbed network. Second, the Red Team could not disrupt services, bring down hosts, and delete files. This rule was introduced to avoid actions from the Red Team that would have jeopardized the effectiveness of the detection tools of the Blue Team.
Participation in the Blue Team/Red Team exercise accounted for 20% of the final grade. The students had to break into sub-teams with specific tasks. At the end of the

exercise, each team had to submit a report. The report format was specified in detail so that the instructor could evaluate a number of parameters, such as the ability to plan in advance both attack and defense, the ability to deploy protection/detection mechanisms and to prepare automated attack scripts, the ability to cooperate with other sub-teams, and the ability to maintain a log of the activities (both attacks and detections).

Setup

The setup for this exercise required the configuration of two sets of hosts, one set for each team. Both teams needed root access to the hosts in order to set up attack and defense tools. The teams were told to prepare and test their tools on the class testbed network and to be ready to move their tools to different hosts right before the exercise. This was done to push student to develop portable software.

It was decided that the Blue Team hosts would be four of the hosts in the class testbed. The IP addresses of the hosts were changed, to make identification of the hosts not completely trivial. In addition, the operating systems on these hosts were re-installed to avoid the possibility of Trojan-ed software left by components of the Red Team. The network was instrumented so that a complete dump of the traffic could be collected.

The Red Team was given privileged access to a set of hosts located in an instructional lab, where the exercise took place. These hosts were the main concern for the administrators, because the students could use their privileged access to attack other hosts in the instructional lab and access the departmental file server. It was decided that the advantage in terms of management overcame the risks, and that the students could be trusted (at least for a four-hour period).

Execution

The exercise included a two-hour preparation phase, where the two teams set up their tools, and a two-hour execution phase, where the actual competition took place.
The day of the exercise, an instructional lab was completely reserved for the exercise. The room was divided into two zones, one for each of the teams.

The preparation phase was carried out without surprises. The testbed hosts were made accessible to the members of the Blue Team, who installed their tools and decided the location of the secret files. The Red Team installed the attack tools on the hosts that were placed in the instructional lab.

When the actual attack phase started, the atmosphere in the lab heated up. The students were very excited and there was a general feel that a competition had started. The competition was not just about getting a good grade in the class. The students actually felt that they were part of a team, and they had a sort of team pride.

The Blue Team had developed a number of network-based decoy tools, which were supposed to confuse the adversaries. These tools were simple but very effective. They ranged from sniffers that would respond to ICMP requests even when directed to non-existent hosts, to tools that would simulate the existence of multiple hosts by 'mirroring' the behavior of one. In addition, the Blue Team created host monitoring software that acted as a form of host-based intrusion detection.

The Red Team also developed a number of tools. Most of them were filters to translate the outputs of scanning tools into a format that was usable by tools developed by other team-mates. The attack process was organized in detail: the attackers had an 'attack pipeline' where the results from one team were given as input to the following team in a continuous process.

During the execution of the attack a few incidents occurred. A couple of times the scanning activity of the Red Team crashed a victim host. The hosts were then rebooted and restored. In a small number of instances the monitoring systems developed by the Blue Team overloaded the monitored hosts to the point that they were unresponsive and, in two instances, they had to be rebooted.

Apart from these events, the exercise progressed smoothly. The Red Team was able to successfully compromise all the hosts and access the secret files. Most of the attacks of the Red Team were successfully detected by the Blue Team. In addition, the decoys and the defense tools developed by the Blue Team successfully slowed down the attackers.

Lessons Learned

- Having a team that is responsible for defending only and a team that is responsible for attacking only has a number of disadvantages. First of all, the members of the defense team think they are having 'less fun' than the members of the attack team. In addition, they feel that protecting and detecting requires much more work than attacking. This last observation was confirmed by comparing the tools developed for the exercise. The Blue Team developed tools that were much more sophisticated than the Red Team tools. This is mostly because of the restrictions imposed on the defenders in terms of network filter configuration and OS patching.
- The development of original tools should be required, or at least rewarded more. The Red Team members downloaded most of their attack tools from the network and concentrated most of their efforts on coordinating the activities of different sub-teams. It would have been preferable to have more of the Red Team's effort devoted to developing new attack tools.
- It is necessary to specify a precise format for both the description of the attacks and the detection logs. The reports from the students contained very imprecise descriptions of both. Often, basic information (e.g., correct timestamps and TCP ports involved) was missing. This made it impossible to correctly match the descriptions of the attacks performed by the Red Team with the detections reported by the Blue Team. In addition, no automated processing was possible.
- It is important to stress the importance of a *process.* Students tend to take shortcuts (e.g., an attempt to run a known exploit blindly against the 255 addresses of a subnet) in order to win the competition. Instead, it is important to foster the preparation and the execution of a well-defined process.
- It is important that the two teams work in different rooms. Having the two teams sharing the same lab space causes a number of problems. First of all, some of the students' energy is devoted to checking if some members of the other team are trespassing. Second, noise and cheering from a team may disturb or irritate the other team.

- The Blue Team and the Red Team need to be on different IP subnets. This makes management and filtering simpler. In addition, by having attackers and defenders separated by intermediate routers it is possible to create a more realistic setup.

A Structured Testbed

Our experience with the flat testbed topology used in the first edition of the class showed that it was necessary to include hardware mechanisms to be able to partition the users of the testbed network. For this reason, the second edition of the class included a redesign of the testbed network. Given the success of the previous edition of the class, it was possible to obtain some funding from the Department to build a new (and better) testbed network.

The structure of the testbed included two separate subnetworks connected by a router. This allowed us to assign separate network addresses to separate teams, solving the problem of interference during the testing phase. The use of a router that separates the two subnetworks also allowed us to use specific firewall rules to limit (and log) the amount of interaction between the two subnetworks. The router was implemented as a multi-homed host connected to each separate subnetwork and to the outside world. The host used Network Address Translation to relay internal traffic to the outside. By doing this, it was possible to use a single routable IP address for the whole testbed, which simplifies considerably its management.

Another innovation in this testbed was the use of an image server to allow for the automatic reinstallation of OS platforms. Images of freshly installed operating systems were saved on a dedicated server. Whenever a reinstallation was needed a host was rebooted using a special disk containing the restoring software, namely Partition Image (Dupoux 2003). The software would connect to the image server and restore a clean image of the operating system in few minutes.

Even though the use of an image server simplified the procedure necessary to restore malfunctioning operating system, and the partitioning of the testbed into two subnetworks created less interference between the class teams, as the time got close to the deadline for the live exercise, the teams started to push the testbed network to its limits. In particular, each team used the hosts in its own subnetwork as both attackers and victims. This situation caused an increase in the number of incidents that prevented the testbed from functioning properly.

A Capture the Flag EXERCISE

This exercise was carried out as part of the second edition of the class. The goal was to modify the Red Team/Blue Team exercise to take into account the lessons learned in the previous editions of the class.

The exercise was organized in a way similar to the Red Team/Blue Team exercise, with the difference that there was an attempt to balance the attack and defense responsibilities between the two teams.

The class was divided into two teams: The Alpha Team and the Omega Team. Both Teams were responsible for both attacking the other team and defending their own assets. More precisely, each team was responsible for protecting a set of hosts and

hiding a flag (the secret file described in the Red Team/Blue Team live exercise) on every host. The team's goal was not to prevent the other team from breaking into the hosts. Instead, the priority was to detect the attacks of the opponents. In addition, each team had to attack the other team's hosts and retrieve the flags for each of the attacked hosts.

The rules that were imposed to the two teams were similar to those described in the previous exercise: the teams could not use *a priori* knowledge about the testbed network; the teams could not disrupt services, bring down hosts, or delete files; they could not filter/block network traffic and/or patch vulnerable software.

Participation in the 'Capture the Flag' exercise accounted for 20% of the final grade.

Setup

The setup for this exercise was different with respect to the original Red Team/Blue Team exercise. Two different instructional labs, one for each team, were reserved for the exercise. The labs were on different IP subnets. Two sets of hosts different from the ones used for the class testbed were prepared and configured in an identical way. In addition, it was decided to connect all the hosts to a hub and to provide extra connection ports for the students' personal laptops. This way they could pre-install some of the attack/defense software prior to the exercise. A complete dump of the traffic directed to the victim hosts was collected.

Execution

When the exercise started, each team gathered in their assigned instructional lab. Then, each team was given the hosts to be protected.

At the beginning of the exercise, the teams had two hours of 'truce' to prepare their hosts for the exercise (installation of attack/defense software, hiding of the flags, etc). The truce was actually enforced by a set of rules in the router connecting the two instructional labs. The actual exercise was carried out in the following two hours.

This time the students were strongly encouraged to develop their own tools. The results were impressive: the students created complete honeypots using virtual machines (e.g., User Mode Linux and VMware) and built very complex attack tools to improve the resilience to decoy techniques.

Lessons Learned

- It is important to push the students to be precise in identifying their targets. This is mainly to prevent attacks from getting out of hand, but also to make them understand the subtleties of stealthy attacks.
- Collecting data during the attack is an important activity. Extra effort should be devoted to collect host audit trails. These are particularly valuable for use in future editions of the class (e.g., audit trail analysis assignments) and as research data.
- The creation of unnecessary traffic during an exercise should be penalized. By penalizing the excessive generation of traffic it is possible to prevent the students from launching massive denial-of-service attacks against the opponents'

hosts and force them to use advanced techniques that use the least amount of traffic.

- It would be beneficial if the students were required to proceed through a path that would force them to progressively make their way to a complex network. The Red Team/Blue Team and Capture the Flag exercises had a 'flat' structure: the same techniques were applied iteratively to a number of targets and there were no changes in the mission's goals during the exercise.

A Network with Victim Hosts

The lessons learned from previous editions of the class suggested an improvement of the testbed network. For the third edition of the class it was decided to add to original testbed configuration a subnetwork that would contain victim hosts only.

Therefore, the new testbed featured a multi-homed host connected to three internal subnetworks and to the outside. Two of the subnetworks were allocated to each of the class teams. A third subnetwork contained a set of hosts that could be used as targets for attacks.

The victim machines were configured in a fail-safe mode, where a process would periodically check their availability, and, in case of malfunctioning, it would restore a clean installation of the operating system and it would execute a reboot.

The rules of the firewall allowed traffic between each of the two subnetworks allocated to the teams and the target subnetwork, but allowed no traffic from the 'target' subnetwork to the outside. This allowed for a more isolated environment for the testing of security attacks, without the possibility of interrupting the research activity of other students and limiting the interference between the two class teams.
We found that this testbed configuration provides the best tradeoff between functionality and ease of management

A Treasure Hunt Exercise

In this exercise, executed as part of the third edition of the class, the Alpha and Omega teams competed in a treasure hunt. The treasure hunt goal was to break into a simulated (yet realistic) payroll system and perform a money transfer transaction.

Each team had to perform a number of tasks (e.g., scan a network or break into a host). Each task had to be completed in a limited amount of time (e.g., 30 minutes). The first team that achieved the task got 5 points. If the other team completed the task within the specified time, it received 3 points. If the time elapsed and the team was not able to complete the task, then a cheat-sheet was provided so that the task could be completed, but no points were given. A task was disclosed only after the previous one was completed by both teams. The list of tasks is presented in Table 1.

In this exercise, no detection task was required (an exercise similar to the "Capture the Flag" described previously was carried out early in the quarter as a form of midterm.). The teams had to concentrate on attack techniques only. The goal was to motivate the students to be prepared for the unknown and to be able to deal with unforeseen problems. In addition, a considerable amount of stress was put on the production of truth files, that is, files that contain a complete specification of the attacks that were

carried out. These files had to be produced in IDMEF (Curry 2003) format, for automated processing.

In preparation for the exercise it was suggested that each team build expertise in a list of topics: network scanning techniques, attacks against SQL servers (both local and remote), NIS-based and NFS-based attacks, buffer overflow attacks (both local and remote), privilege escalation techniques, password cracking techniques, attacks against Apache web servers, attacks against CGI applications. This list was provided to focus the energy of the teams on techniques that would be useful during the exercise.

Task	Description	Max Duration
1	Determine the active hosts in subnet X.Y.Z. Also determine each host's OS and the services/applications that are remotely accessible. Scanning techniques that will evade detection by the Snort system will receive additional bonus points.	20 minutes
2	Get interactive access to the web server host by exploiting a web-based vulnerability. You must be able to login into the host as a user account other than root.	30 minutes
3	Get root privileges on the web server host.	30 minutes
4	Determine the hosts that are located in the specified internal subnet. Also determine their OSs and the services/applications that are remotely accessible. Scanning techniques that will evade detection by the Snort system will receive additional bonus points.	20 minutes
5	Access the MySQL database on host SQL and obtain the content of the table Employees.	20 minutes
6	Get interactive access to the MySQL server host. You have to be able to login with an account that is not root.	20 minutes
7	Get root access to the MySQL server host.	20 minutes
8	Modify the database table Employees, setting the account number of each employee to an account number of your choice.	10 minutes
9	Obtain access to the transaction service on host TRN. Schedule a paycheck payment that will transfer the employee paychecks to your account.	30 minutes

Table 1: List of tasks used during the Scavenger Hunt exercise

It was made very clear to the students that the ultimate goal of the exercise was to perform a multi-step attack that was as realistic as possible. One of the lessons learned from previous exercises is that the data collected during these exercise is valuable from both the instructional and the research viewpoints. The traces collected in the previous exercises lacked an underlying ‘plan’. That is, it was desirable to have traces

of attacks that had a well-defined final goal. This is particularly useful for alert correlation purposes. Therefore, an important by-product of this exercise was the Tcpdump data, BSM data, Windows event logs, and Snare events collected on the networks and hosts used during the exercise. Combined with the truth files produced by the students, these traces are invaluable resources for researchers in the field of intrusion detection and attack correlation (Lindskog 1999).

Setup

In order to prevent the two teams from interfering with each other, two identical target networks were setup. The topology of the networks used in the exercise is shown in Figure 1. In the following, we describe a single target network.

The web server was placed on a DMZ network. The MySQL server (host SQL), the file server (host NFS), and the transaction server (host TRN) were placed on a separate network, accessible only by the web server host.

The web server was an Apache server, running as user apache, as per default installation. In addition to a fake corporation site, a number of broken CGI scripts were installed on the web server. One CGI script was vulnerable to a phf-style attack. Another CGI script contained information about how to log into the MySQL database, namely a clear text password. A program for checking the syntax of perl files was 'erroneously' left around in the CGI directory. This program could be invoked through the web server. The program allowed one to view the source code of all the CGI scripts installed on the server, disclosing important information, such as embedded MySQL passwords.

The file server was configured to export the file system /home to the world. This is a security mistake often present in internal networks, where security is more relaxed. In addition, the host NFS was configured to provide password files through NIS.

The MySQL server provided remote access to user dbuser with password bsecret. Note that by default MySQL allows local access to root without having to provide a password. The server mounted the /home file system from the file server.

The transaction server had a service running on port 7979. The transaction application was developed *ad hoc* for this exercise by the instructor.

When connecting to port 7979, the user was dropped into a simple shell application. Typing HELP would show a list of commands, one of which is PAYCHECK. The team was supposed to invoke that function to transfer all the employee paychecks to the attacker's account. That function required a password. The encrypted (but very guessable) password was stored in the password file distributed through NIS.

This setup required a considerable effort. I developed the web site, the code for the CGI scripts and some applications, and the SQL schema for the database with the help of the teaching assistants and some of my PhD students. In addition, the networks had to be physically set up, and a whole set of services were created and configured on each network.

The setup of the exercise and the testing of the network configuration required two days of work for a team of four people.

Execution

The day of the exercise the two teams gathered in two separate (but nearby) instructional labs. The execution of the exercise included a first hour where the teams would prepare their tools, followed by a three-hour period during which the actual treasure hunt was performed.

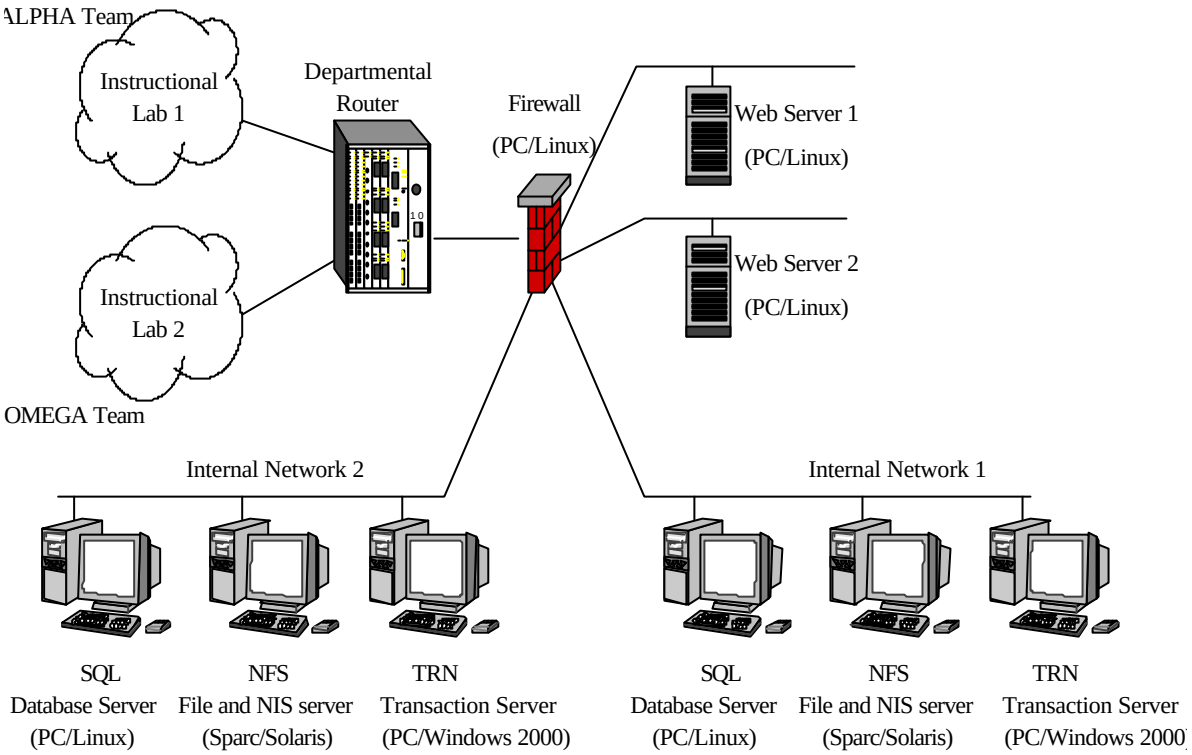

Figure 1. Treasure Hunt: Network setup

The students were extremely excited about this exercise. In this exercise there was no direct clash of teams. Therefore, there was no fear that a team could do something illegal to jeopardize the mission of the other team. By the same token, the exercise was structured as a race: the first to achieve a given task would get the most points. This motivated the students to organize their sub teams effectively.

The exercise progressed seamlessly up to task seven, where one of the team wasn't able to complete the task and had to use a cheat sheet. The same team also had significant problems in performing the final task and needed to be helped (so that the exercise could finish). These difficulties were due to the way the teams were created.

The Alpha Team was composed of students whose last names were between A and I, while the Omega Team was composed of students whose last names were between J and Z. This division didn't take into account the skills of the individual students, and by chance the most skillful students ended in the Alpha Team.

Lessons Learned

- Setting up separate targets for the two teams and having them race against each other is a very good way to foster competition without having to deal with the less pleasurable aspects of a direct clash between the teams.
- Building balanced teams is important. It promotes a fair and interesting exercise, and at the same time it supports the morale of the students by letting them know that there is not a 'best' team.
- Collecting traces of all the actions performed by the students is important. This activity should not be limited to network traffic, but it should also include host-based audit trails (e.g., Windows events).
- It is important to educate the students to create *good* truth files. These are extremely useful to identify the attacks within the logs.
- This type of exercise requires twice the effort needed to set up exercises like those described in previous sections.

Related Work

The use of hands-on experience in labs is obviously not new. Testbed networks have been previously used to provide root-level access to students. A different approach is used by a class taught in Stanford (Boneh 2002) where students can experiment with their techniques on isolated virtual machines. While this approach gives a reasonably precise idea of the working of security attacks, it is different from the 'real thing.' Only a testbed network can provide a realistic environment.

Even though hands-on experience is advocated by many, there are few graduate and undergraduate courses on computer security that offer live exercises as part of the course. For example, Georgia Tech offers a class (Santos 2002) where a team has to install a number of services on a Windows host and another team has to perform attacks. In this case, each of the two phases, preparation and attack, lasts a couple of weeks. As another example George Washington University offers a class (Daniel 2002) that includes the creation of honeypots and some sort of team-based interaction.

In general, live exercises are difficult to organize and conduct, and, therefore, instructors generally prefer other types of educational tools, which are less expensive in terms of time and hardware/software resources. We believe that our experience, especially the adoption of the treasure hunt exercise is rather unique.

Conclusions and Future Work

Testbed networks and live exercises are important instructional tools in teaching the practical aspects of network security. Testbed networks provide a safe, isolated environment where students can experiment with potentially dangerous technologies. Live exercises motivate the students to give their best because of the competitive nature of the test, and because their success is heavily determined by the students' creativity.

Testbed networks and live exercises are also extremely difficult to organize and manage. In particular, testbed networks require constant management and the implementation of effective recovery techniques. Live exercise require detailed preparation, because they are executed in a short time span and if anything goes wrong it is difficult to solve problems within such tight schedule.

This paper described the testbed networks and the live exercises that have been used as part of a graduate-level course on network security and intrusion detection. The class has a successful history of attendance (the maximum allowed is 40, but classes ranged between 50 and 90). The success of the class is determined by the practical, hands-on approach that was adopted. The student feedback was overwhelmingly positive. Some of the students that took a previous edition of the class even came to observe the live exercises of other editions.

This class has received some attention by other instructors. The class materials have always been online and have been used in other courses (with permission of the instructor). In addition, the by-products of the class attracted the interest of research groups. We are currently preparing a web site to distribute the traces collected during the different exercises, in addition to the course material. We hope that this effort will allow other courses to use our experience to build similar testbeds and live exercises.

The next step in the evolution these live exercises will be the creation of an inter-institutional 'Capture The Flag' exercise, where students that are attending similar practical security classes at different universities can test their skills against each other. This form of live exercise will be very challenging because of the high risk of creating unfair conditions and the possibility of a very high level of competition. For this reason, the careful design of the exercise is critical. We plan to use the lessons learned from the design and implementation of previous live exercises to design effective mechanisms for containment and measurement of this experiment.

References

Bishop, M. (1997). The State of INFOSEC Education in Academia: Present and Future Directions. In *Proceedings of the National Colloquium on Information System Security Education*, pages 19–33. Keynote address, Linthicum, MD, April 1997.

Bishop, M. (1999). What Do We Mean By "Computer Security Education"? In *Proceedings of the 22nd National Information Systems Security Conference*. Arlington, VA, October 1999.

Bishop, M. (2000). Academia and Education in Information Security: Four Years Later. In *Proceedings of the Fourth National Colloquium on Information System Security Education*. Washington, DC , May 2000.

Boneh, D. (2002). *CS155: Computer and Network Security.* Stanford University.
Brandt, A. (2003) Class on Virus Creation Draws Industry Ire, *PC WORLD*, May 2003.

Curry, D. and Debar, H. (2003). Intrusion Detection Message Exchange Format: Extensible Markup Language (XML) Document Type Definition. IETF Proposed Standard, Version 1.0, http://www.ietf.org/html.charters/idwg-charter.html, January 2003.

Daniel, R. (2002). *ECE 297 - Special Topics. Network Security: Honeypots.* The George Washington University, School of Engineering and Applied Science, Department of Electrical and Computer Engineering.

Dupoux, F. (2003) *Partition Image*, URL: http://sourceforge.net/projects/partimage. June 2003.

Lindskog, S., Lindqvist, U., and Jonsson, E. (1999). IT Security Research and Education in Synergy. In *Proceedings of the First World Conference on Information Security Education (WISE1)*, pages 147--162, Kista, Sweden.

Santos, A.D. (2002). *CS6265 Information Security Lab*. Department of Computer Science, College of Computing, Georgia Tech.
Schneier, B. (2003) *Crypto-gram Newsletter*, June 2003.

Certification and Accreditation: A Program for Practitioner Education

K. L. Burke[1], C. W. Rasmussen[2], C. E. Irvine[3], G. W. Dinolt[4], and T. E. Levin[5]

[1]Computer Science Department
Naval Postgraduate School, Monterey, California, USA,
E-Mail: klburke@nps.navy.mil;

[2]Mathematics Department
Naval Postgraduate School, Monterey, California, USA,
E-Mail: ras@nps.navy.mil;

[3]Computer Science Department
Naval Postgraduate School, Monterey, California, USA,
E-Mail: irvine@nps.navy.mil;

[4]Computer Science Department
Naval Postgraduate School, Monterey, California, USA,
E-Mail: gwdinolt@nps.navy.mil;

[5]Computer Science Department
Naval Postgraduate School, Monterey, California, USA,
E-Mail: televin@nps.navy.mil;

Abstract

Large complex systems need to be analyzed prior to operation so that those depending upon them for the protection of their information have a well-defined understanding of the measures that have been taken to achieve security and the residual risk the system owner assumes during its operation. The U.S. military calls this analysis and vetting process certification and accreditation. Today there is a large, unsatisfied need for personnel qualified to conduct system certifications. An educational program to address those needs is described.

Introduction

Computer and network systems process information critical to enterprise security. Should these information systems be vulnerable to security failures or attacks, the consequences could be grave. Although individual components may provide security features and assurance of correct policy enforcement, their encompassing systems and subsystems are frequently large and complex. How can a system owner assess the suitability of a system to operate in a particular environment? Factors that will affect this determination include the sensitivity and criticality of the information to be processed; the physical and cyber context in which the system is expected to operate; the personnel who will administer and use the system; as well as a wide variety of technical factors that affect security.

The process used to assess networks and systems and to then officially authorize their use is known as *certification and accreditation*. For example, an avionics system might be the

ISSN 1445-3312 print/ISSN 1445-3347 online

subject of a certification and accreditation. *Accreditation* is a formal declaration by a designated approving authority that an Automated Information System (AIS) is approved to operate in a particular security mode using a prescribed set of safeguards (NSTISSC 2000a).

In general, accreditation will result in the approval for the system to be operated with defined physical conditions, interconnections, personnel security attributes, and system assurances, in combination with procedural and technical countermeasures to security threats. A threat is any circumstance or event with the potential to harm an IS through unauthorized access, destruction, disclosure, modification of data, and/or denial of service (NSTISSC 2000a). The accreditation describes the operational objectives of the system, defines the threats to the system and the countermeasures taken to mitigate those threats, and the resulting residual risks. As part of the process it is recognized that a reassessment of system security is required periodically, so the accreditation will have a limited lifetime.

Certification is the comprehensive evaluation of the technical and non-technical security features of an AIS and other safeguards, made in support of the accreditation process, to establish the extent to which a particular design and implementation meets a set of specified security requirements (NSTISSC 2000a). As the system moves through its lifecycle, the certifier works with component designers and integrators to ensure that a specified set of security requirements is met.

System Certification and Accreditation (DoD 1997) can help to identify and mitigate risk in a wide variety of systems. Consequently, the U.S. Department of Defense (DoD) has stated that all information systems will be certified and accredited to operate at an acceptable level of risk. Given the sheer numbers of systems in operation, from business systems to weapons system, this is a daunting task.

It is clear that a highly skilled cadre of system certifiers is needed, not only to address the current demands of the government but also to provide similar support for the complex systems being fielded in the private sector. Yet, there are relatively few analysts with the background, training and education that would qualify for senior leadership positions in system certification. To address the gap between requirements and available qualified personnel, we are establishing an educational program for system certifiers.

Herein, we provide a high-level overview of the certification and accreditation process using the U.S. DoD certification and accreditation model as our example. We will then describe the program we are developing to provide certifiers with the education and experience needed to progress from a beginner to an intermediate level.

Certification and Accreditation

To ensure that all services perform accreditations to some standard level, the DoD has published an instruction called The DoD Information Technology Security Certification and Accreditation Process (DITSCAP) (DoD 2000a). The DITSCAP was designed to be a flexible standard process, readily tailored to support C&A efforts on a variety of systems including acquisition, legacy, locally-acquired, and deployable systems. This instruction process provides a degree of confidence that all accredited systems have undergone an equal and adequate level of analysis and testing. Realistically, however, the outcome of certification and accreditation is dependent on the education and experience of the personnel conducting the exercise. Qualified personnel are in short supply, and the need for individuals to provide technology support for Certification and Accreditation will continue to grow.

The following sections provide a brief summary of the information system certification and accreditation (C&A) process defined in relevant instructions and publications (DoD 1997, DoD 2000, DoN 2000a, DoN 2000b, DoN 2000c). We have chosen to focus on Navy requirements and our overview is intended to illustrate the complexity of the C&A task, and the fact that the transition from apprentice to journeyman certifier requires training, formal education, and field experience.

Who is Involved?

There are four principal participants in the C&A process:

Program Manager (PM). The Program Manager is the individual responsible for system procurement and development, operations, or maintenance, depending upon life cycle stage (DoN 2000a). According to the DITSCAP, 'program manager' might refer to three distinct roles over the life of a system. During system acquisition, the program manager is the individual responsible for system procurement and development. During the operation of the system, the role belongs to the system manager, who is responsible for system operations. When the system undergoes a major change, the role belongs to the maintenance organization's program manager.

Designated Approving Authority (DAA). The DAA is the official with the authority to formally assume responsibility for operating an AIS or network at an acceptable level of risk (NSTISSC 2000a). It is the DAA who is ultimately in the position of accepting an inevitable compromise between the desire for perfect security, the minimum set of security features required by applicable legal or regulatory constraints, and the needs of the user community to have a functional system that meets its needs. It is the DAA who assumes the risk; only upon accreditation by the DAA does the system become operational and able to run with 'live' data.

Certifier. Either alone or as a member of a team, the system certifier provides a comprehensive evaluation of the security features, limitations, and vulnerabilities of a target information system. It is the certifier's responsibility to document for the DAA the target system's level of compliance with security requirements and the level of residual risk present in putting the system in operation. Residual risk is the amount of risk remaining after security measures have been applied (NSTISSC 2000a)

User Representative. This individual requires that the system in question achieve a specified level of functionality.

Functional Components of Certification & Accreditation Process

This section provides an overview of the functional components of the Certification and Accreditation process. By appreciating this process, the role and contribution of the System Certifier can be understood in context.

The DITSCAP process is divided into four major phases: Definition, Certification, Validation, and Post-Accreditation. Table 1 provides a synopsis of the steps that must be accomplished during each phase. The DITSCAP process may be iterative and for large, complex systems it is sometimes necessary to conduct several iterations.

Phase	Step	Description
Definition		
	1	Document Mission Need
	2	Conduct Registration
	3	Perform Negotiation
	4	Prepare System Security Authorization Agreement
Certification		
	5	Support System Development
	6	Perform Certification Analysis
Validation		
	7	Certification Evaluation
	8	Develop Recommendation to Designated Approval Authority
Maintenance		
	9	Compliance Validation
	10	Maintenance of System Security Authorization Agreement

Table 1: Functional Components in the Certification and Accreditation Process

Definition

This phase comprises the first four steps discussed in this document: documentation of mission need, registration, negotiation, and preparation of the System Security Authorization Agreement (SSAA) (this step is often incorporated into the negotiation step).

Document Mission Need

This preliminary phase occurs whenever development of a new information system or modification of an existing system is initiated. Planning the certification begins with acquiring a thorough understanding of the system to be certified, the functions that the system must fulfil, and the mission served by the system. This planning also requires a comprehensive understanding of the steps required in all C&A processes. The certifier keeps all concerned personnel fully informed even at this early stage in the process. Of particular importance are the following:

- Proposed system mission.
- Proposed system functions.
- Proposed system interfaces.
- Category and classification of information to be processed.
- Anticipated system lifecycle.
- Characteristics of system users.
- Operating environment.

System Registration

The registration phase is the beginning of the dialogue among the key players in the C&A process. The steps vary, depending on whether the subject system has been fielded previously or is under development. The first step in the registration phase is a review of the materials from either a new Document Mission Need phase or from a previous life cycle iteration. The final step in the registration phase is the development of a draft (or draft update) of the SSAA. In either case, the draft SSAA represents an agreement among the Program Manager, the DAA, the CA, and the user representative, and describes the goals that must be achieved in support of certification as well as the strategy by which those goals are to be met.

The following list describes key steps in the process.

- Register the system: Inform key participants (DAA, Certifier, PM, User representative) that the C&A process must be undertaken.
- Prepare mission description and system identification. In the case of a new system, this step relies on the documentation developed in the previous step. In the case of a system that has already been in operation, this step relies on the body of documentation, including the existing SSAA that should accompany the system throughout its life cycle.
- Describe the system environment and threat description. The system environment has both physical and logical components. For example, a locked cage in a guarded room presents a much different picture from the standpoint of vulnerability than does a desktop in a busy office. Similarly, a stand-alone system presents a much more difficult target than, for example, a system with multiple network connections or connection to the Internet.
- Describe the system architecture and C&A boundary. This boundary describes precisely which equipment and systems within the domain of the DAA are to be subjected to the C&A process under development.
- Determine the IT system security requirements. Minimum security controls are mandated by the DoD, and can be strengthened (but not weakened) by the military services.
- Prepare a DITSCAP plan based on the required documentation. Based upon the preceding steps, this step tailors the DITSCAP tasks to the system under consideration.
- Identify organizations and additional resources required for the C&A process; this step facilitates measurement of the level of effort that will be required.
- Develop the draft SSAA. This document constitutes the basis for the negotiation phase, which follows.

Perform Negotiation

In the negotiation phase all parties have an opportunity to express their needs and agree on their respective responsibilities. The principals agree on strategy, resources, roles, timeline, etc. In reality, the certifier might, for example, have to convince a user representative that allowing users to hold administrative privileges is unacceptable, or persuade a DAA the level of residual risk claimed by the certifier. The draft SSAA resulting from the registration phase provides a framework for the negotiations. The DITSCAP identifies three key negotiation tasks:

- Review the draft SSAA for accuracy and completeness, updating as necessary.
- Conduct a review of the certification requirements, modifying the SSAA as necessary.
- Approve the SSAA, which constitutes the blueprint for the balance of the certification process.

Prepare the System Security Authorization Agreement (SSAA)

The SSAA encompasses in a single document all essential security-related information about a system. It includes the product of the steps accomplished in the Definition Phase. As a living document, the SSAA is still subject to updates at every subsequent step prior to accreditation.

The principal components of the SSAA are:

- Mission Description and System Identification. Much of this can come from the mission needs statement. Of interest are the system name and identification, the physical and functional descriptions of the system, and a summary of the system concept of operations.

- Description of System Operating Environment. This encompasses technical and non-technical context in which the system will be operated, software, and maintenance environments, as well as a threat description.
- Description of System Architecture. This comprises hardware, software, firmware, interfaces, information flow, and accreditation boundary.
- System Security Requirements. These, including national and DoD/DoN requirements, data security requirements, security concept of operations, network connection rules, configuration and change management requirements, and re-accreditation requirements.
- Organizations and Resources Required for the C&A Effort. This item identifies the principals (PM, DAA, Certifier, User Representative) and sponsoring organization, enumerates staffing and funding requirements, certification team training requirements, describes roles and responsibilities, and identifies any additional organizations or groups whose participation is required.
- The DITSCAP Plan (tailored as necessary). This includes tailoring specifics, tasks/milestones, the schedule of work, level of effort, and specification of roles and responsibilities.
- Appendices containing supporting and/or amplifying documentation (e.g. policy, security concept of operations, etc.).

Certification

This phase comprises the next two steps: support of system development and certification analysis.

Supporting Systems Development

This is the first step in the Certification Phase of the DITSCAP, concerned with verification that a system that is in development remains compliant with the security specifications of the SSAA. This requires more or less continuous oversight on the part of the Certifier as system development and/or integration progresses. The precise details are determined by a number of factors, including the certification level specified in the SSAA and the position of the system in its lifecycle, e.g., new system development or system maintenance. Education in the area of computer and network security is essential in this part of the certification process. The NSTISSI #4015 certifier training document (NSTISSC 2000b) identifies the following performance items associated with this step:

- Coordination with Related Disciplines. This involves coordination with various security disciplines for expert assistance. For example, it might be necessary to call in experts on physical security, or emanation security, or cryptography. The certifier needs to justify to the DAA the need for such coordination, and to ensure that the coordinated effort is successfully accomplished.
- Configuration Control. The certifier must evaluate configuration and change control with regard to consistency with requirements, recommending changes and/or reporting deficiencies as necessary. Included in this step is verification of associated activities, such as audits, component inventories, etc.
- Information Security Policy. The certifier must identify all applicable information systems security policies, keeping the development team fully informed in order to enable system compliance. The certifier must also monitor development to ensure compliance.
- Life-Cycle System Security Planning. The certifier must evaluate the life-cycle security plan adopted by the development team. If the plan is deficient, the certifier must become an active participant in life-cycle security planning to ensure the desired outcome.

- Principles and Practices of Information Security. The certifier must understand the principles and practices of information security and the way in which those principles apply to the certification effort in question. The certifier must also adhere to these principles and, if necessary, explain these principles to the development team.
- Network Vulnerabilities. The certifier must perform system analysis to identify potential network vulnerabilities for the development team, evaluate the potential impact of such vulnerabilities, and suggest corrective measures.

Perform Certification Analysis

The certification analysis step determines whether the system in question is ready to advance to the evaluation and testing that precede a recommendation to accredit. The DITSCAP specifies the following component tasks:

- System Architecture Analysis. This task verifies that the system architecture is consistent with the architecture agreed on in the SSAA.
  - Security architecture is evaluated to ensure it is consistent with specified security policy and requirements.
  - Interfaces between the subject system and other systems are identified and evaluated in terms of supporting the required system security posture.
- Software Design Analysis. The output of this step documents that security features required of the Trusted Computing Base (TCB), such as authentication, access control, and auditing, are implemented as specified. (The TCB is the suite of security features interacting within a given information system to enforce a specified security policy.)
- Network Connection Rule Compliance Analysis. This step provides assurance that neither the network nor the subject system will have undesired effects on the other's security posture.
- Integrity Analysis of Integrated Products. The subject system might integrate software, hardware, and firmware from a number of sources, e.g. commercial-off-the-shelf, government-off-the-shelf, specialized, etc. This step provides assurance that:
  - Interaction of integrated components does not result in degradation of the integrity of individual components.
  - The result of this integration is compliant with the specified system security architecture.
  - Application of components is consistent with their intended use. The complexity of this step can be considerable, depending upon the level of certification required. For example, it might be necessary to verify the security features of individual components.
- Life Cycle Management Analysis. This step provides documented assurance that the security posture of the system will be preserved by the implemented change control and configuration management practices.
- Vulnerability Assessment. This step verifies satisfactory progress in implementation of the security requirements of the SSAA, by evaluating vulnerabilities and recommending countermeasures. Any vulnerability identified during certification analysis must be analyzed in terms of susceptibility to (and likelihood of) exploitation, and of the associated threat. The output of this process is a statement enumerating and evaluating residual risks and estimating the operational impact of accepting or rejecting them. Residual risk cannot exceed the level of acceptable risk determined by the DAA.

Validation Phase

Like the Certification Phase, the Validation Phase also comprises two steps: certification evaluation and development of the recommendation to the DAA culminating in accreditation.

Certification Evaluation

The objective of this step is to ensure that the system, configured for deployment, complies with the security specifications as given in the SSAA. Certification evaluation is applied to hardware, software, firmware, and additionally includes site inspection. The main functional items are listed below.

- Security Test and Evaluation
- Penetration Testing
- TEMPEST and Red-Black verification
- Validation of COMSEC compliance
- System management analysis
- Site accreditation survey
- Contingency plan evaluation
- Risk-based management review

Develop Recommendation to DAA

In this activity the Certification Authority (the Certifier) i.e., the manager of the certification process) submits to the DAA a report detailing all findings from the certification process and makes an accreditation recommendation to the DAA. If the process has been successful, the DAA formally accepts the (positive) recommendation and the outcome is accreditation. If change is required, an Interim Approval to Operate may be granted and, all or part of the certification effort is revisited. The following elements are identified:

- Access Control Policies. Access control policies implemented in the system to be certified must be explained to the DAA. Included in this explanation are descriptions of who makes authorization decisions and on what basis as well as the effectiveness of the implementation from the standpoint of the requirements. The certifier recommends changes, if necessary.
- Administrative Security Policies and Procedures. The certifier must consider not only those policies and procedures required by law, but also those additional policies and procedures that might be required by agency instruction or other organizational mechanism. The certifier must document to the DAA all applicable policies and procedures and the degree to which the system is in compliance, recommending countermeasures as needed to address any deficiencies.
- Certification. This is a conditional recommendation, outlining (if necessary) conditions that must be met before a decision to accredit is recommended.
- Presentation of Security Test and Evaluation Results. The objective is to communicate the results to management and technical personnel.
- Identification of Potential Corrective Approaches
- Determination of Residual Risk

Post-Accreditation

Finally, the Post-Accreditation Phase corresponds to ongoing maintenance of the SSAA.

Compliance Validation

At intervals specified in the SSAA, the system and its operational environment are subject to review to verify compliance with the SSAA in terms of security specifications and concept of operations, and to verify that the threat assessment described in the SSAA remains accurate. The principal functional components are:

- Physical security analysis
- Review of SSAA with an update to the SSAA as needed
- Risk-based management review
- Procedural analysis
- Compliance re-verification

Maintenance of the SSAA

While the SSAA is subjected to continuous review and update during system development, the maintenance step outlined here occurs post-accreditation to ensure that the SSAA continually reflects the operational system. The principal players are the same as they have been throughout the process. As the operational system undergoes incremental change, the certifier evaluates the impact of these changes on system security features, updating the SSAA, if necessary. Updates must be evaluated in order to determine whether the Certification process must be repeated. If so, the process reverts to the appropriate DITSCAP phase. The certifier ensures that the DAA has up to date information, and the DAA will determine whether continued operation of the system is approved. Key components in this step are:

- Control of Configuration Changes
- Maintenance of Configuration Documents
- Periodic Review of System Life-Cycle
- Contingency Planning
- Compliance Validation
- Physical Security
- SSAA Review
- Risk-based Management Review
- Compliance Re-verification

Certifier Education

A considerable amount of technical and non-technical analysis is required to support an accreditation. This process of system certification provides a way by which the technical and non-technical aspects of a system's security can be assessed from its inception through retirement. The factors that must be addressed include the sensitivity and criticality of data to be processed, the system's environment, its users, its location, its applications, interconnections, configuration, etc. To achieve these objectives, such activities as security test and evaluation, risk analysis, and a variety of other analyses and evaluations are conducted. The level of technical expertise required for individuals involved in certification is high. Even while focussing on a single security component of the system, the certifier must keep the larger system context in mind and be able to understand the impact and side effects of that component on overall system security. Thus the certifier cannot address his or her task simply by using a checklist at the end of the process, or by focussing on individual pieces, while neglecting the whole.

As is the case with many other aspects of computer science and system development, e.g. construction of operating systems or construction of physical databases, one does not learn everything in books or in a standard classroom. Even laboratory activities can be inadequate unless they are specifically designed to foster the development of both implicit as well as explicit knowledge. In the case of system certifiers, it has been found that a combination of knowledge and experience are essential for achieving mastery of the profession.

To address this problem, we have developed an educational program for certifiers. It is intended to compress the time it takes an apprentice certifier to achieve the experience and expertise to become a journeyman certifier. We believe that master certifiers are those individuals who have considerable experience and have the education, knowledge and fully internalized skills to assess the security properties of highly complex systems. In a sense the activities of the certifier parallel those of a systems integrator. Just as there is no expectation that a highly experienced systems integrator can be created through a set of classroom activities, there is no expectation that a master certifier can be produced in such a manner.

Students in the program will be of two types: short course students and resident graduate students. Short course students will typically be personnel who may already be working in the area of C&A or who are moving into this field. The resident students will be active-duty officers, or civilians employed either by the DoD or by DoD contractors. In all likelihood, graduates of the short program will eventually report to graduates of the resident program. The short program students will spend approximately eight weeks in formal courses over a period of from eighteen months to two years. The courses will be of short duration (typically two weeks) and high intensity, with eight hours devoted to class and laboratory exercises each day. The intervening periods between visits to school will be spent in the field, where students acquire essential experience. Resident students will include certifier courses as electives as part of their Computer Science graduate program. Depending upon student background, validation of prerequisites, and other factors, this program can last between 12 and 24 months. The certifier courses taken by the resident students will differ from those taken by the short-program students. The courses taken by the resident students will be taught in the usual way, meeting five hours per week over an entire academic quarter. Course content might also differ somewhat in reflection of the different educational and career paths taken by the two populations of students.

A prerequisite for all resident students is an undergraduate degree in computer science or a closely related engineering field.

The courses are briefly described in Table 2. The first three courses are intended to provide students with an understanding of the problem domain for system certification. Introduction to Information Assurance is a survey course and provides students with a broad overview of the many aspects of the certification domain. The second course, Secure Management of Systems, leads to an understanding of the administrative, procedural, and personnel issues that might affect the ongoing security of a system. Finally, Network Security Threat Analysis provides students with an appreciation of the techniques and skills that will be brought to bear by adversaries attacking their systems. When combined with their background in computer science the three-course sequence described above prepares students for the two courses specific to certification.

Introduction to Certification and Accreditation is intended to teach students about all aspects of the C&A process. They are introduced to procedural aspects of the process as well as to the variety of technical issues that might be addressed. A considerable amount of social skill and team building is required for a successful certification, and students learn about the give-and-take required to achieve success. Students must understand when certain security requirements must be adhered to at all cost and when some flexibility may be appropriate.

The capstone course in the sequence centers on a group of case studies. These are taken from real systems and allow students to understand how a certifier can help ensure that the security

requirements are met. The cases include not only technical and procedural aspects of the certification, but discussion of the social process required accomplishing the certification.

An unusual aspect of the program is its mentoring process. Students in the program will have the opportunity to interact with instructors and staff who have experience in DITSCAP certification. This mentoring experience will help speed their mastery of the certification process. A member of our educational team with significant experience in certification keeps in touch with short course students while they are in the field gaining on-the-job experience. Students can communicate and commiserate with each other about their challenges and experiences. Because the certifier community is relatively small, it is expected that students will get to know senior certifiers and be able to ask them questions as they progress.

Program assessment will be a feedback mechanism that should benefit from the involvement of the sponsoring organization. The sponsoring organization is one of the principal Navy commands involved in certification and accreditation of IT systems and components. Student assessment will be to some extent program-dependent. The performance of resident students will be assessed in the usual ways, by examination scores, performance on laboratory exercises, quality of written work, etc. The performance of the non-resident students will be based not only on their classroom and laboratory performance while here at NPS but also on their performance on the job between visits to NPS. Both populations will be assessed on their abilities to apply the regulatory framework (e.g., DITSCAP) to systems that vary widely in their makeup. Students in both populations will benefit from success factors that are built into the program. For example, the students will arrive on board with appropriate backgrounds, the material covered will be chosen with the assistance of experienced professionals from the field, and case studies will include both system-level and component-level case studies.

Title	**Catalogue Description**
Introduction to Information Assurance: Computer Security (CS 3600)	Provides a comprehensive overview of the terminology, concepts, issues, policies, and technologies associated with the field of Information Assurance. It covers the notions of threats, vulnerabilities, risks and safeguards as they pertain to the desired information security properties of confidentiality, integrity, authenticity and availability for all information that is processed, stored, or transmitted in information systems.
Information Assurance: Secure Management of Systems (CS 3670)	Provides students with a security manager's view of the diverse management concerns associated with administering and operating an automated information system facility with minimized risk. Students will examine both the technical and non-technical security issues associated with managing a computer facility, with emphasis on DoD systems and policies. Students will earn CNSS (formerly NSTISSI) certification for: INFOSEC professional, Systems Administrator, and ISSO.
Network Security Threat Analysis (CS3675)	This course is designed to give the student exposure to Internet security threats in a lab environment. Lectures and labs provide the student with a 'hands on' experience with current network attacks and vulnerabilities. Foot-printing, scanning, enumeration and escalation are addressed from an attack prospective. Emphasis on detection and protection of critical data and nodes is addressed. A final project that demonstrates skills and knowledge is required.

Title	Catalogue Description
Introduction to Certification and Accreditation (CS4680)	This course provides an introduction to the Certification and Accreditation (C&A) process as applied to procurement and lifecycle management of DoD and Federal information systems. Topics include: principal roles, functional components, and output documents of the C&A process; and a comparison of the government C&A process specification currently in use (DITSCAP/NIACAP, FIPS) with the emerging effort to produce a unified specification.
System Certification Case Studies (CS4685)	This course is the second part of the Certification and Accreditation course sequence (CS4680 and CS4685). Students will investigate 2-3 case studies of systems that have been evaluated, and then apply the lessons of CS4680 to make final accreditation decisions. Successful completion of this two-course sequence along with CS 3600 and CS 3670 leads to NSTISSI DAA and Certifier certification.

Table 2: Courses of the Certifier Education Program

Two surveys will be used for requesting feedback from the non-resident graduates and their 'on-the-job' mentors. When the non-resident students graduate from the course, they will go to certification organizations as certifiers. In most certification organizations, the new certifiers are teamed with experienced certifiers for their initial certifications. These experienced certifiers act as their mentors. Generally these initial certifications are on the less complex systems that require a lower level of certification. As the new certifiers gains experience, they undertake increasingly more complex or secure systems. These progressions occur with the approval of the experienced certifier/mentor, until eventually the certifier is considered experienced enough to certify alone.

The senior mentors will be asked to complete a survey, giving us feedback on whether or not the mentor feels the certifier had enough classroom training and what areas need to be modified or added. To assess our program, we will ask the new certifiers to evaluate how well the certifiers' course prepared them for their certification experiences. Again, we would welcome suggestions for improvement. Also, we will maintain a continuing relationship with graduates, offering them continued mentoring. Not only will this feedback loop assist us in assessing the certifiers' course, it will assist us in ensuring that the course material reflects current systems.

Summary

Large complex systems should be analyzed prior to operation so that those depending upon them for the protection of their information will have a well-defined understanding of the measures that have been taken to achieve security and the residual risk the system owner assumes during its operation. The U.S. military calls this analysis and vetting process certification and accreditation. Today there is a large, unsatisfied need for personnel qualified to conduct system certifications. We have described an educational program designed to address those needs.

Acknowledgements

The authors wish to thank the United States Navy, in particular CNO, N6 and the Space and Naval Warfare Systems Command, for support that has made this work possible.

References

DoD (1997) DoD Instruction 5200.40, Department of Defense Information Technology Security Certification and Accreditation Program (DITSCAP)

DoD (2000).DITSCAP Application Manual, DoD 8510-1-M

DoN (2000a) DoN IA Publication 5239-13, Volume I: Introduction to Certification and Accreditation

DoN (2000b) DoN IA Publication 5239-13, Volume III: Program of Record Information Systems

DoN (2000) DoN IA Publication 5239-13, Volume II: Site, Installed Program of Record, and Locally Acquired Systems

NSTISSC (2000a) National Information System Security (INFOSEC) Glossary, NSTISSI No. 4009

NSTISSC (2000b) National Training Standard for System Certifiers, NSTISSI No. 4015

Security Dimension of IT in Developing Countries: Risks and Challenges

Respickius Casmir[1], and Louise Yngström[2]

Department of Computer and Systems Sciences (DSV)
Stockholm University/Royal Institute of Technology, Sweden
E-mail: [1]si-rc@dsv.su.se and [2]louise@dsv.su.se

Abstract

In the present paper we discuss security aspects of the Information and Communication Technology (ICT) from developing countries perspective. Specifically, we present and discuss our study done in Tanzania. We start by discussing the extents of computers and computer systems use and network connectivity, followed by level of IT security awareness and the country policies in relation to ICT implementations. Then we highlight various risks and challenges within the context and finally, we discuss some of the remedial steps and actions that are being taken to deal with the situations.

Introduction

Today most of developing countries are striving to build up their economic capacities, moving from the predominance of the agricultural sector into a competitive and dynamic knowledge-driven economy. While some countries are in the midst of a transition from the industrial to the service economy, others are undergoing a transformation from a planned to a market-based and demand-driven economy. The vast majority of these countries are striving to acquire and make use of Information and Communication Technologies (ICT) to bring down what is termed as the *digital divide* (NTIA 1999). In this case Tanzania being one of the developing countries is no exception. New technologies are introduced continually, and existing ones become obsolete almost as soon as they appear. However, as the Internet and networked systems become more widespread and advanced, there is a higher risk of accidents, attacks, and failures. Thus, a proper empowering of the citizens to exploit the advantages of ICT has to be coupled with appropriate public awareness on Information Technology (IT) security issues underpinning a sustained and concerted development effort for the country.

In this paper we present our study and experience with Tanzania as a case study. Our discussions are focusing on extents of computers and computer systems use, network connectivity, IT security awareness and the country policies in relation to ICT. Then we highlight various risks and challenges within the context and finally, we discuss some of the remedial steps and actions that are being taken to deal with the situations.

Computer and Computer Systems Usage

Information and Communication Systems are increasingly becoming integral to modern culture and are the primary engines behind much of the world's economic growth. The field of Information and Communication Technologies (ICT) continues to evolve at an astonishing pace. This is true to Tanzania as it is elsewhere around the world. The rapid evolution of the

ISSN 1445-3312 print/ISSN 1445-3347 online

discipline has a profound effect on culture, education, and the business affairs. It is also affecting both communication and interactions between and among people.

Proliferation of Internet Cafés

One of the differences between Tanzania and other Southern African Development Community (SADC) countries is the proliferation of Internet cafés, particularly in Dar es Salaam but also elsewhere. It is very difficult to give the exact figure for the number of Internet cafés in Tanzania because they are mushrooming on a tremendous scale. However, the number was estimated to 1000 plus as of November 2001 (Lamtrac 2001) most of them in Dar es Salaam, which is the business capital of Tanzania. Retail internet charges in most cafes in Dar Es Salaam recently dropped from roughly US$1 to US $ 0.5, that is equivalent to Tshs. 1000.00 and 500.00 respectively per hour, but this is still enough to meet the operating costs and also make a little profit. The bulk of clients in most cafés are young people, business people, office workers, students and academics. E-mail services are the most popular at Internet cafés, followed by general website surfing. While a number of people go to make telephone calls abroad which is cheap via the net, others go to the cafes on e-business missions but the proportion in this category is relatively small.

Pervasiveness of the World Wide Web

Pervasiveness of the World Wide Web in the country is increasing from day to day. For instance, if you turn on a radio or television or open a magazine or newspaper, chances are that you will hear, see or read an advert or story about the web. Today, most organisations in the country, including state agencies and local governments, maintain websites. The proliferation of Internet cafés coupled with numerous downloadable web contents makes it even more attractive to every individual who happen to have access to the net. Training centres teaching basic computer skills and how to use the Internet are mushrooming and are always full booked. People are in queues waiting to join into these courses. E-Commerce is also picking up at higher pace. People are enjoying P2P (peer-to-peer) file-sharing technology in the Internet.

Today the benefits have been enormous. For example, the government is now using Internet technology to enhance the access to and delivery of government services to benefit citizens, government's business partners and employees. Using the Internet to complement traditional means of service delivery, such as written communication, telephone, fax and counter services, provides greater scope for citizens and businesses to access government services in a timely and efficient manner.

Network Connectivity

As of March 2002, Tanzania Communication Commission (TCC) (TCC 2003) had issued licences to seven local companies to provide public data communication services in Tanzania. These include Wilken Afsat, Datel Tanzania, Equant Tanzania, Simbanet Tanzania, SoftTech Tanzania, Fastcom Africa and Tanzania Telecommunication Company (TTCL). The licence essentially allows these companies to provide infrastructure for use by other service providers, for example Internet Service Providers (ISPs). With the first Cellular phone operator starting late in 1993, today there are five active cellular phone operators in the country. These include Mobitel, Tritel, Celtel, Vodacom and Zantel. The latter is mostly dominant in the Isles part of the country, Zanzibar. As of March 2001 Mobitel was leading among others with about 124000 subscribers followed closely by Vodacom with about 93000 (Esselaar 2001). According to Esselaar's report the growth in number of cellular subscribers has been dramatic in recent years (estimated to 65% p.a. from 1998 to 2000).

Internet Service Providers (ISPs)

As of May 2002, there were thirteen licensed ISPs in Tanzania. Nearly all of these ISPs offer two types of Internet services, Dial-Up services and wireless communication services. The latter is mainly for corporate clients. The number of Dial-Up accounts was estimated to more than 15000 in November 2001 (Esselaar 2001). Despite higher telephone tariffs, the number of new users subscribing for Dial-Up services is highly increasing. The monthly subscription fees for Dial-Up account are ranging from US$30 to US$50 exclusive of telephone charges during the connection session. Individual users are not opting for wireless service because of higher initial cost for the wireless equipment.

International Connectivity

Currently, all data providers in Tanzania have access to the Internet via satellite communication links, VSAT. Thus, bandwidth is one of the obstacles to Internet access to the vast majority of people in the country. Available bandwidths are ranging from 64kbps to 1Mbps. In most cases this is a shared bandwidth amongst more than one ISP resulting into rapid saturations as the number of users increase at a given time. However, projects that are at various stages of completion might significantly reduce the problem of the international bandwidth. These include the following:

SAT-3/WASC/SAFE project

Commissioned in May 2002, SAT-3/WASC/SAFE cable is co-owned by a 36-member consortium formed for construction of the submarine optic fibre cable system (SAT-3 2002). This state-of-the art cable system connects Europe with Africa and Asia. Going around Western Africa, South Africa and India from Portugal in Europe before terminating in Malaysia in the Far East, this cable provides enhanced capacity, diversity and connectivity to all the consortium members. The recently commissioned SAT-3/WASC/SAFE submarine cable has been operational since May 27, 2002. The cable system has an ultimate capacity of 120 Gbps that will enable it to convey a total of 5.8 million simultaneous telephone channels. The world's first undersea optic fibre cable system around Africa to Europe and Asia was, officially inaugurated by the Senegalese Head of State Mr Abdoulaye WADE.

Although Tanzania is not part of this phase, the opportunity to connect to the system exists. There is already a project backed by Transnet, Telekom South Africa, the rail group Africa East Coast and others to lay 2500 KM of optic fibre from Dar es Salaam along the existing railway line to Livingstone in Zambia (Esselaar 2001). The plan is then to link this with a fibre optic cable laid by Namibia Telecommunications from Windhoek to Livingstone, with connections to other east and central African countries planned (Kenya, Uganda, Burundi and Rwanda). Windhoek already has a high capacity fibre optic link to South Africa that can then connect to SAT-3.

IT Security Awareness

Our study revealed that with all these promising steps in terms of ICT development, the country with nearly 30 million inhabitants had neither an academic institution nor a training centre offering IT security program or courses. As a result each organisation, institution and individuals were implementing IT security in their own way. We further learned that in most organisations IT security was part of traditional security, that is, the focus was on the security of physical or tangible assets. Computer hardware, peripherals, active and passive network equipment were of paramount when it comes to IT security in most organisations. Most of people we interviewed believed that this physical security measures were adequate to secure

data and information that are being processed, in transit and/or stored within computer systems. None of the 64 organisations we investigated had an IT security policy in place. Likewise neither of these organisations had a training program for its employees on IT security awareness.

Country Policies in Relation to ICT

For a period of nearly 20 years, Tanzania had experienced an ICT dormant era (Lamtrac 2001). This was between 1974 to early 90's. During this period policies governing importation of computers to the country were so strict and bureaucratic in such a way that few people could manage to own a computer. However, in the early 90's the Tanzania's government realised the need to move into the world of ICT and amended its policies on the issue. The government further realised that computers were no longer a luxury but a necessity. Today Tanzania is one the SADC countries that are in forefront to address and take advantages of ICT to build its economy.

Bold steps are being taken to leverage the benefits of Information and Communication Technologies (ICT) for its national priorities of growth and poverty alleviation (Accenture 2001). Key actions include: the creation of an e-secretariat (eSecretariat 1998), including key stakeholders to create supportive leadership for ICT development communications infrastructure improvements, both in the capital city of Dar es Salaam and in secondary towns. Also, early in 1999 the Tanzania Planning Commission published the Tanzania Development Vision 2025 (Vision 1999). The Vision comprises the top-level national guidelines that direct government activities.

Tanzania's Vision 2025 singles out ICT as a key driver for transformation to realise competence and competitiveness. Specifically, the Vision states that, ICTs are major driving force for the realisation of the Vision. As it prepares for the shocks associated with the transition from an agriculture-based economy to a knowledge-based economy, Tanzania hopes to illustrate that starting off on the right foot is the key to leapfrogging or antelope-jumping many stages of ICT development (Accenture 2001).

In May 2002, the government released the first draft of the national ICT policy in its history (MoCT, 2002). The recently released national ICT policy draft, however, does not stipulate how to deal with security issues, especially, the training and education part of it. In general terms, information security is not given the attention it deserves in Tanzania. Also, the government has waived taxes on computers imported to the country since year 2001.

Security Risks and Challenges

Thanks to the Government of Tanzania for realising the importance and the need to foster information and communication technology (ICT) for its economic development. However, our study has revealed that there is yet a challenge to the government and citizens at large that is not addressed so far. This is the security of data and information stored in, processed and transmitted within computers and computer systems. Also, the security for users of IT systems' such as their privacy and the system's infrastructures is not adequately addressed. Observed challenges or unaddressed security issues in the country include the following:

1. No national security standards and/or guideline for various IT products and systems
2. No central public unit responsible to receive, respond and keep records of computer misuse reports. Such unit, should it exist, could also be used to alert people on various foreseeable cyber threats

3. No policies to address software piracy
4. No policies to protect kids against cyber threats
5. Most of the law enforcers are not only IT security illiterate but also computer illiterates
6. Most kids are more knowledgeable in computer use than their parents and schoolteachers
7. The vast majority of practitioners are working with computers handling sensitive data and information such as medical records without even a basic knowledge of IT security
8. No single institution is offering IT security training/education in the country

These are just a few to mention since the list was endless. During the study we also observed the following security risks.

Children and the Internet

With all numerous benefits of the Internet and the World Wide Web, there are associated risks that might cause some adverse effects to users of the net if extra care is not taken. For instance, the study has revealed that in all the Internet Cafés in the country there ware no security policies, restrictions or guides for children under 18 in using this facility. You find primary school kids flocking to these Internet cafés during their leisure surfing the net on their own. The problem here is that neither parents nor the kids themselves are aware of the security risks involved on surfing the net. Neither the café's operators ware well aware of such risks. Also up to the present time of the study there is no country policy with regard to children and the Internet use.

While the Internet is fundamentally a great place for children, there are so many areas of cyberspace that are not appropriate, just as there are areas in almost every city that are inappropriate for children. There are also certain activities on the Internet that may be appropriate for adults but not for children, and areas that are suitable for some children and not for others. That is why it is important to train and educate parents to help them decide what areas are right for their own children. Of course, a right thing is subjective, since how a certain web site or activity looks from within one family is different from the way it is viewed by the whole society or a whole country or even a whole world for that matter. In any case, parents need to know that Internet is a global medium therefore things like child pornography are common on the web.

Adults and the Internet

Security risks are not limited to kids only. The risks might be even higher for adults. For instance, with increased connectivity and extensive use of the World Wide Web in the country, the study has revealed that most of grownup people tend to believe in all information they find on the web. On this we have a vivid example where a grownup person who happened to be a graduate electrical engineer, lost his US$500 as an application fee for graduate study sponsorship. One day he was browsing the web searching for institutions offering some financial sponsorship to study abroad when he found a site requiring him to send application for sponsorship through the web at a fee of US$500. The fee had to be sent immediately to the specified bank account for his application to be considered. There were a lot of attractive promises in the sponsorship package such covering of the tuition fee, meals and accommodations, transport, stationeries and some stipend allowance. The deadline for application was set so close in such a way that he could not think of verifying such

information with colleagues. Up to now he had never received any reply and the web site does not exist anymore.

Similar incidents have happened, and are continuing due to low level of security awareness to the majority. There are a lot of other social engineering activities in the Internet that are affecting the vast majority of the society. People are giving a lot of personal information to some sites in the Internet without knowing the ultimate use of such information. Some are even giving out their company's crucial business information unknowingly. Some websites hosted in the country are not secured enough. This is very dangerous since some attackers might exploit such vulnerabilities and use them as a jumping stone to attack others even by masquerading themselves as someone else.

Viruses and other Malicious Codes

Viruses are seriously affecting the majority of people frequently in Tanzania as elsewhere. Some of the victims are absolutely unaware of the existence of antivirus software, and some are aware but do not know where to get it from or how to implement it. Some people are aware and have antivirus software installed in their computers but they are not aware that it needs to be updated regularly. Generally, people need some sort of training on how to deal with viruses and other presumptions malicious codes such as deleting without opening email attachments with **.exe** extensions and any other attachments from people they do not know.

Password Basics

Most users use passwords in order to be granted access to their computer systems. It was learnt, however, that most users tend to trust their passwords too much. They are absolutely unaware that passwords can be spoofed or cracked with some freely downloadable software from the net (RPC 2003). Even worse, some of these users prefer to pick one password, and 1) use it for all of their accounts, 2) use it all the time, 3) never have to change it, and 4) write it down so that they can reference it if they happen to forget it. The problem here is that if the password is easy to remember, it is easy to guess. If the password is written down, guessing does not even matter. And if the password is never changed, then repeated attacks are more likely to occur. Many people also think that it is inconvenient to have a timeout feature on their computers. They do not understand that such features are for security purposes, to lock their workstations so that when they are away from their desks other malicious users cannot impersonate and have unauthorised access to their computers systems.

Social Engineering Acts

In its simplest terms social engineering is, basically, the art and science of getting people to comply with your wishes. The aim of such acts is to trick people into revealing 1) their password or other information that compromises a target system's security, 2) credit card numbers or other personal information that compromises the individual, or 3) to gain access to unauthorised areas where equipment is stored. Most people in the country are not aware of these acts. The fact is that all computers and computer systems rely on humans. Humans turn on computers, create passwords, enter credit card numbers, and manage these systems. Therefore, the human part of security is independent of platform, software, hardware, networks, and technology used. People need to be trained and educated on various forms of social engineering so that they do not fall into this trap.

Backup of Data

It has been revealed that most practitioners in the country do not commonly practice creating backups for their sensitive data. With exceptions of a few financial and business institutions

such as banks, insurances, and some government institutions, most organisations do not create backups of their data. In some cases, it was learnt that backups of sensitive data is done and stored in the same computer room as the original data. This is so dangerous especially in case of catastrophic disasters such as fire, floods, and theft, which are common problems in developing countries

Security Risks of Peer-to-Peer Software across the Internet

Over the past years there have been developed a number of programs for peer-to-peer communications across the Internet (Senior and Deters, 2002). The first, perhaps best known, of these was *Napster*, which was used across the world to exchange MP3 (music) files. Other programs developed since then have offered file sharing for more general types of files. A non-exhaustive list of these programs would be: Napster, Aimster, Gnutella, Morpheus, KaZaA, Edonkey2000, Napigator, Limewire, Bearshare,and WinMX. The use of such software, especially by teenagers, in Tanzania is increasing at an alarming rate.

These programs can be used for professional purposes. However, their usage in Tanzania as it is in the general world tends to be for retrieving and sharing audio (MP3) and video files and pirate versions of commercial software. Such use may raise issues of legality, and in particular copyrights. Also, as for any software downloaded from the Internet, these programs can introduce security risks such as viruses, Trojan horses and backdoors. In addition, these programs may contain spyware which can violate confidentiality. *Spyware programs* collect information about the user and usage of the computer (email addresses, types of Web sites visited and so on). They may also cause extra network traffic by regularly popping up browser windows to particular network sites.

Unfortunately, most of people who are using these peer-to-peer programs are unaware of security risks they are facing. This suggests establishment of an awareness program on IT security related issues not only in Tanzania but also in other developing countries with ICT situations similar to Tanzania.

Summary of Risks and Challenges

In summary, all the aforementioned issues are simply IT security basics that need to be adequately addressed. However, these are not the only ones but just a few to mention. There are a lot more security tips and tricks that Tanzanians and other people in the developing countries need to know, especially, end users most of which are non security professionals, and yet common users of computers and computer systems in their day to day operations.

Internet is an educative, informative, and entertaining facility. However, at times it can turn into a destructive facility if adequate security precautions are not taken timely. Internet security risks are not to be taken lightly especially in this part of the world where people are so enthusiastic with this new and fantastic facility. People in this region tend to adopt, use and trust any technology from abroad without testing its reliability. While it is true that these risks cannot be eliminated completely, it is also true that they can be managed and minimised just like any other security risks in business and in general life.

We need to train and educate people about IT systems' threats and risks, and how to deal with them. We believe that setting up security over the Internet is mainly a matter of knowledge. People need to know about hackers and attackers, other cyber criminals and their intentions, and how to deal with them. They also need to know about proactive, preventive, detective,

and reactive measures and techniques for different types of threats. Therefore, well informed people can best safeguard their systems and keep them within acceptable limits of risks.

Remedial Actions Being Taken

To the best of our knowledge, one of the best ways to handle such situations is to train and educate people on IT security issues. To this end, we have developed an undergraduate IT security Curriculum Model that will be implemented at the University of Dar es Salaam. The curriculum can also be implemented at any other interested academic institutions in any developing country as well as in any other country. Also, plans are underway to establish an IT security Unit within the University of Dar es Salaam that will handle IT security related issues for the University and for the general public at large. It is scheduled to launch the Unit by November 2003.

Conclusions

Although the paper discusses situations that are tailored to the realities of Tanzania, we believe that most of developing countries, especially, sub-Saharan countries with exception of South Africa faces more or less similar situations. One of the reasons for this belief is that people from some Universities in countries like Kenya, Uganda, Rwanda, and Burundi are usually coming to the University of Dar es Salaam to for consultations on ICT implementation issues.

References

Accenture, Markle Foundation, & United Nations Development Programme (UNDP). (2001) Creating a Development Dynamic – Final Report of the Digital Opportunity Initiative: A Global Call to Action. URL: http://www.opt-init.org/framework/pages/contents.html [Date accessed: 24 May, 2003]

Anderson, R. J. (2001) Security engineering: *A Guide to Building Dependable Distributed Systems*, John Wiley & Sons, Inc.

eSecretariat, Tanzania. (2000) URL: http://www.ethinktanktz.org/esecretariat/Contactus.htm [Date accessed: 13 January, 2003]

Esselaar, Miller and Associates. (2001) *A Country ICT Survey for Tanzania Final Report for Sida*,.URL: http://www.sida.se/Sida/articles/9400-9499/9481/tanrep.pdf [Date accessed: 19 March, 2003]

Senior, M., Deters, R. (2002) Market Structures in Peer Computation Sharing, *IEEE-P2P. The Second IEEE International Conference on Peer-to-Peer Computing: Use of Computers at the Edge of Networks (P2P, Grid, Clusters) Conference Proceedings*, Linköping, Sweden.

KPMG. (2002) *East Africa Fraud Survey*. URL: http://www.kpmg.co.ke/other/East%20Africa%20Fraud%20Survey%202002.pdf [Date accessed: 22 May, 2003]

Lamtrac AB. (2001) *Survey of the Need for a Vocational Training Programme for ICT Professionals in Tanzania Final Report*. URL: http://www.ethinktanktz.org/esecretariat/DocArchive/Final%20report%20IT-Survey.pdf [Date accessed: 12 April, 2003]

MoCT, Ministry of Communications and Transport. (2002) *National ICT Policy of Tanzania* (First order draft [v3.2e]). The United Republic of Tanzania URL: http://www.moct.go.tz/ict/zerothorder.pdf [Date accessed: 9 February, 2003]

NTIA, National Telecommunications and Information Administration. (1999) *Falling through the Net: Defining the digital divide*. Washington, DC: Department of Commerce.

Pfleeger, C. P. (1997) *Security in Computing*, 2nd Ed..Prentice-Hall, Inc., Upper Saddle River, NJ.

Power, R. (2002) *Computer Security Issues and Trends*, 2002 CSI/FBI Computer Crime and Security Survey, **VIII**,1, (Spring).

RPC, Russian Password Cracker. (2003) URL: http://www.password-crackers.com [Date accessed: 15 May, 2003]

SAT-3/WASC/SAFE Project. (2002) URL: http://www.safe-sat3.co.za/ [Date accessed: 10 May, 2003]

Schneier, B. (2000) *Secrets and Lies: Digital Security in a Networked World*, John Wiley, New York.

TCC, Tanzania Communications Commission. (2003). URL: http://www.tcc.go.tz/ [Date accessed: 17 March, 2003]

Teemupekka V. (2002). Four View on Security, *PhD Dissertation*, Helsinki University of Technology, (pp. 7-31).

Vision, Planning Commission. (1999) *The Tanzania Development Vision 2025*, The United Republic of Tanzania. URL: http://www.tanzania.go.tz/vission_2025f.html

Enhancing Information Warfare Education Through the Use of Virtual and Isolated Networks

D.J. Ragsdale[1], S. D. Lathrop[2], and R. C. Dodge[3]

[1] Information Technology and Operations Center
United States Military Academy, West Point, New York,
E-mail: daniel.ragsdale@usma.edu;

[2]Information Technology and Operations Center
United States Military Academy, West Point, New York,
E-mail: scott.lathrop@usma.edu@usma.edu;

[3]Information Technology and Operations Center
United States Military Academy, West Point, New York,
E-mail: Ronald.dodge@usma.edu;

Abstract

The Information Analysis and Research (IWAR) laboratory at the United States Military Academy (USMA) has proven to be an exceptional and necessary resource for educating our cadets and faculty studying information warfare. The laboratory has also been successful in motivating the need for continued education and training in this area on a much larger scope that touches the highest levels of our military and government. This paper justifies why information warfare laboratories are necessary, explains the laboratories design and organization, and describes the phenomenon that is occurring as a result of the IWAR laboratory.

Introduction

Three years ago, the Information Technology and Operations Center at the United States Military Academy (USMA) developed the initial Information Warfare Analysis and Research (IWAR) laboratory to support undergraduate education and facilitate faculty research in Information Assurance (IA). Since that time, it has matured into a robust, mobile laboratory used to educate all of our students at West Point at different levels of technical submersion. It is also used as a focal point for congressional, academic, military, and other distinguished visitors interested in learning more about Information Assurance, Information Warfare (IW), or in replicating our work. Without the hands-on exposure to the attackers' tools and, through usage, an in depth understanding of the strengths and limitations of our technical defensive counter-measures, we would not be providing a quality and relevant educational experience for our students or leading-edge applied research to our colleagues.

What began as an ad-hoc collection of dumpster bound computers networked together with a few idle hubs and switches, has matured into a world class research and teaching facility consisting of a heterogeneous collection of computer hardware, operating systems, services, and applications networked together to form a production like system but configured specifically to enable the study of both offensive and defensive cyber manoeuvres. Additionally, through the use of VMware, virtual networks have been created on every student's computer thereby allowing the student to administer their own network components. Each laboratory component has a distinct purpose but all are aimed at

ISSN 1445-3312 print/ISSN 1445-3347 online

furthering education in Information Assurance at USMA and throughout the community in general.

The intention of this paper is to provide our motivation for building such a laboratory, discuss related solutions, give an overview of the current state of the laboratory and the methodology used to obtain this state, and provide observations of the increased participation and awareness in IA/IW at both USMA and beyond.

Motivation

It can be argued that education in information warfare is paramount for the students at the United States Military Academy. Nearly a year ago, the Secretary of Defense summarized a long-standing national discussion when he stated that our dependency on information networks makes attractive targets for new forms of cyber attack (Garamone, 2001). In a recent Department of Defense Report to Congress, the assertion was made that "In the future, the network will be the single most important contributor to combat". Furthermore, the report asserted the *information domain* must be protected and defended in order to generate and sustain combat power in the face of offensive actions taken by an adversary (Department of Defense Report to Congress,) With the military's increased reliance on information systems the reasons for educating our students in information warfare are readily apparent.

Consider the fact that the Sapphire Worm infected 90% of the vulnerable servers within 10 minutes, doubling in size every 8.5 seconds, and effecting more than 75,000 systems within 30 minutes (Moore et al., 2003). Had even one percent of those computer systems been military end systems rather than commercial or home-based computers, the effects would have been to cripple the unit's reliance and trust on such systems. If such systems are denied service, or worse the confidentiality or integrity of the data undermined, then clearly information dominance is completely undermined. The future of the military's information superiority on the battlefield hinges on the security of the networked information systems providing the necessary services—thus, the increased requirement that future officers become aware of such issues and their potential solutions.

The issues in assuring our information are much larger than just what the military foresees. Our nation's critical infrastructures and economic structure are becoming increasingly reliant on information systems and the Internet that provides connectivity between such systems. Addressing these issues requires an education in information warfare that does not merely theorize and describes such concepts. A hands-on, *active learning* experience entails that we provide an environment where students, employees, and anyone managing or administrating information systems can apply theoretical concepts in an isolated environment (Felder, 1993). Such an environment allows the unleashing of viruses, worms, and Trojan horses so as not to have an effect on a production network. Kaucher and Saunders found that even for management-oriented graduate courses in Information Assurance, a hands-on, laboratory experience enhances the students understanding of theoretical concepts (Kaucher and Saunders, 2002). The above reasons justified the original creation of the IWAR laboratory and validate continued expansion and improvements to the laboratory.

Laboratory Design

Our requirements for the IWAR laboratory were that we have a facility for our IA course and related courses with the ability to port the work to other courses and research areas, a network for the Cyber Defense competition, a network for our ACM Special Interest Group for Security, Audit, and Control (SIGSAC) student chapter, dedicated systems where users could

locate and download exploits posted to hacker websites, and the reference material necessary to build and maintain the laboratory. The design goals consisted of having heterogeneous operating systems, networking equipment, defensive security tools, and offensive exploits; contain "soft" and "hard" targets; be large enough to provide a real world signature; and be robust enough to withstand the attacks from students and faculty—that is we wanted to make the laboratory conducive to exploitation experiments without creating a lot of administrative overhead in repairing the network. Additionally, we wanted the laboratory to give each student the opportunity to configure and maintain an information system from the perspective of a network and a server administrator. We selected open-source security tools to allow students to "look under the hood" and identify what each tool is doing. What evolved were four separate networks: (1) the IA network with our Virtual Information Assurance Network (VIAN) solution embedded within the student's workstations, (2) the Cyber Defense network, (3) the SIGSAC network, and (4) a small "search box" network. Additionally, we created a library with reference material gathered during the design of the current laboratory. The remainder of this section will focus on the main components of the laboratory.

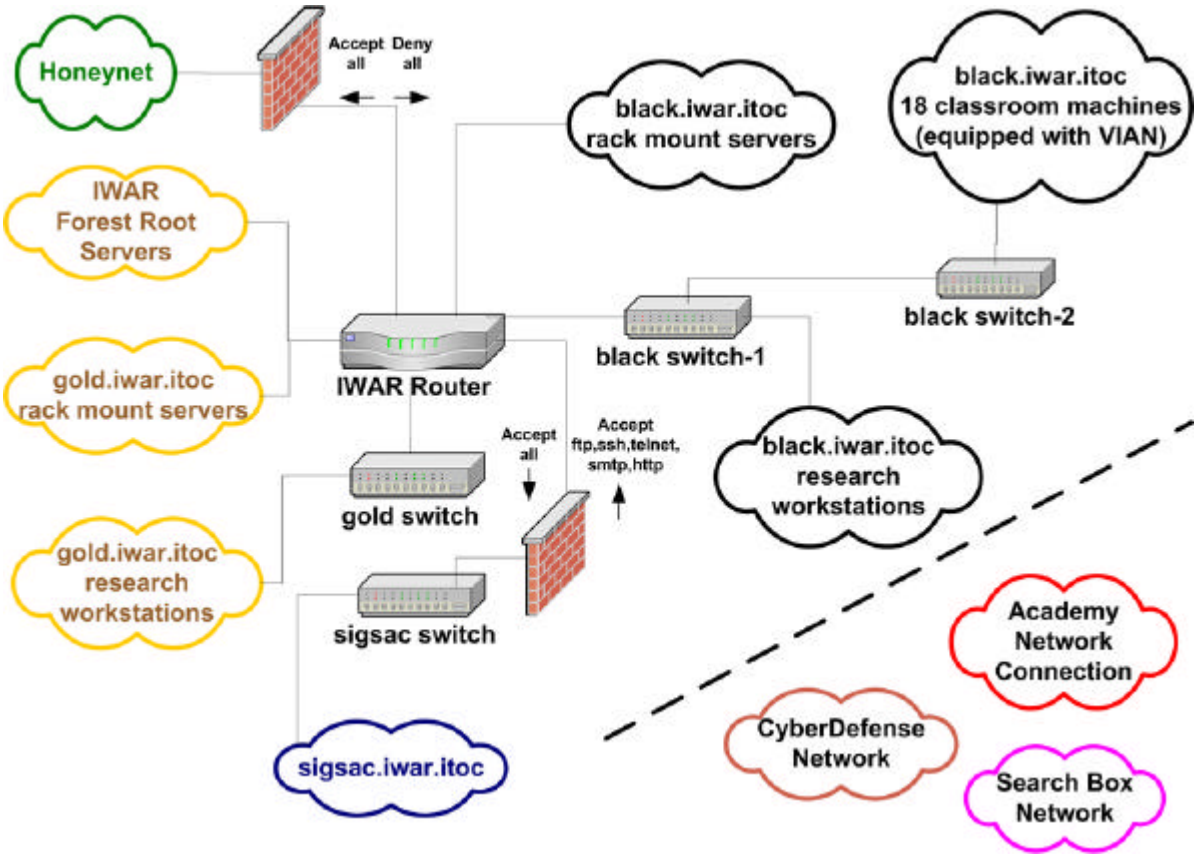

Figure 1: IWAR Laboratory

IA Network

As all of our networks within the laboratory, the IA network is a completely isolated network that we often compare to a firing range. The Army uses firing ranges to train soldiers on individual weapons and firing systems. Likewise, the IA network is a range where students and faculty experiment with both offensive and defensive tools without running the risk of releasing malicious code onto our production network or into the "wild". Just as a solider would only fire a weapon on the range or in combat, the IA network policy only permits users to use the malicious tools in the controlled confines of the laboratory.

The upper right hand portion is the IA network. Table 1 highlights the hardware and operating systems within the IA network. The network contains two major subnets, *black* and *gold*, a honeynet used for research purposes and the SIGSAC subnet used by cadets who are members of our student SIGSAC organization. The *black* subnet contains the classroom machines used for instructional and demonstration purposes; administrative servers; and several *soft* targets such as web, mail, file, and telnet servers. *Soft* targets are defined as machines that have default installations of operating systems and services and/or have older software that has not been appropriately patched. The only "hardening" that has been done to these machines is to insure that all local and domain administrative passwords are strong. The *black* network also has an IEEE 802.11b wireless basic service set (BSS) infrastructure. The wireless network is tied into the *black* network with a wireless access point. Currently there are two laptops with wireless cards and a personal digital assistant (PDA) device with a wireless card used in the network.

Manufacture	Model	CPU	RAM	Hard drive	Operating System	Purpose	Number
DELL	Optiplex GX260	1.8 GHZ	1 GB	80 GB	Windows XP with VIAN	Classroom/ VIAN	20
DELL	Dimension 4500	2.53 GHZ	1 GB	120 GB	Windows XP, Linux, OpenBSD	Research Workstations	6
DELL	Poweredge 1550	1.0 GHZ	512 MB	2, 9GB SCSI	Windows 2000, Linux	Rack mount servers	16
DELL	Optiplex GX300	833 MHz	256 MB	30 GB	Windows 2000, XP, Linux	SIGSAC Workstations	6
Sun	Ultra 5	333 MHz	512 MB	9 GB	Solaris 2.7	Server	2
Apple	Power Macintosh G4	1 GHZ	256 MB	60 GB	MAC OS X	Workstation	1
Sun	Ultra 10	440 MHz	512 MB	9 GB	Solaris 7.2	Server	1
Sun	Sparc Station 20	50 MHz	256 MB	9 GB	Solaris 2.5	Server	1
Sun	SunBlade 100	500 MHz	1 GB	15 GB	Solaris 9.0	Server	2

Table 2: IA Network Hardware

The *gold* network contains our *hard* targets and four research workstations. *Hard* targets are those systems that have been patched and services are up to date. There are various operating systems (Windows NT 4.0, Windows 2000, Windows XP, several Linux distributions, OpenBSD, Mac OS X, and various Solaris versions). The *gold* network is where small groups of students and faculty working on research projects normally operate because of the added security and less risk of losing their work. However, everyone understands to back up their work and store any important files on a zip drive, CD-ROM or on the file server that is off limits to attack. For example, a workstation on the *gold* network has a group of students working on plug-ins and a Java based client for the Nessus vulnerability scanner. This gives the laboratory a "real-world" look-and-feel and also provides the individuals working on their projects some additional security.

Malicious software that exploits system vulnerabilities is stored on an administrative file server to which the students have access. Through experimentation with malicious software, users gain an appreciation of the numerous vulnerabilities existing in currently deployed

information systems. With this knowledge IWAR laboratory users are better equipped to protect and defend the information systems.

We attempt to provide an "enterprise" appearance to the users of the network. For example, within the *black* subnet we have a two web servers running on the various machines. The course web page is on an Apache web server running on a Linux operating system and another web server is running on a Windows based Internet Information Server (IIS) 5.0 server. The Windows Exchange 2000 Server is used not only for email within the IA network, but also as a "soft" target. Other services are added as required by the instructors, students, or other users of the laboratory.

Virtual Information Assurance Network (VIAN)

One of our priorities for the IWAR lab was to give each student the ability to experience, hands-on, the full authority and responsibility of configuring a functional network as both a network and a server administrator. While the isolated IA network gives the student an opportunity to observe reactions on the target network and machines to scan and/or exploit, they are unable to alter or set up the configuration on any of the servers on the network or control the routing and firewall policies. A related issue is that after executing exploits that may cause significant damage to a server, the target machine may need to be rebuilt by an instructor who cannot always be in the laboratory when the students require assistance. For example, when a student releases a buffer overflow exploit against a Microsoft IIS Web Server, that sever becomes ineffective until a reboot. The VIAN concept allows us to offer complete administrative control over a network to the student without placing it or other students' work at risk!

Figure 2: VMWare

VIAN is part of the standard image on each classroom computer. The implementation uses the commercially available VMware Workstation software package (Figure 2). VMware Workstation allows you to create one or more virtual machines that run within a secure virtual boundary on a base operating system. It provides a layer between the host operating system and the guest systems that provides an interface to all physical hardware

resources. The amount of memory resources (disk and RAM) that a guest operating system is assigned and the processor priority is established when the guest operating system is installed. This segmentation provides a completely separate environment for each guest operating system. When the guest operating system is not running, the CPU and RAM resources are returned to the host operating system. (VMWare, 2001)

You can run VMware on a host machine running Windows NT4, 2000, and XP, as well as Linux. The virtual machine (guest) operating system can be any variant of Windows, DOS, Linux, FreeBSD, or OpenBSD. Solaris can be loaded as well, but it has to be run on a raw disk partition rather than a virtual hard drive (a simple file). You can load as many guest operating systems as you have disk space for, and they can be run side by side if enough RAM memory is available on the host machine. Given our hardware configuration for the classroom machines shown in Table 2, our implementation runs six virtual machines simultaneously with no significant latency problems. The 1 GB of memory is extremely important as each virtual machine runs in its own virtual address space.

VIAN, shown in Figure 3, segments the virtual network into two parts. The *red* network contains the attack virtual machines, while the *blue* network has the defensive virtual machines. An iptables based Linux firewall, configured with an "allow all" policy by default, separates the *red* network from the *blue* network. A second iptables firewall separates the *blue* network from the physical *black* network in the IWAR laboratory. The second firewall performs network address translation so that all packets sent to the physical network appear to originate from this firewall. By configuring the second firewall in this manner, allows us to create static IP addresses within the virtual network providing the ability to create one classroom image. Except for the second firewall, the virtual network on each student's workstation is exactly the same. The second firewall receives its external IP address by DHCP thus minimizing administration overhead. If it is necessary to recover a virtual machine because a student configured it improperly or launched an exploit causing irreversible damage, then a simple copying of the virtual machine's files to the host machine is all that is required.

The host operating systems is Windows XP professional. Because we teach two sections of the Information Assurance course, we have installed two separate installations of Windows 2000 server and Red Hat Linux 8.0 on the *blue* network so each user has their dedicated set of servers to administer. Additionally, there is a Windows 98, Windows NT4.0 Server, and Linux Red Hat 6.0 virtual machines installed on the *blue* network for additional targets. The *red* network, consisting of a Windows XP and Linux Red Hat 8.0 instance, is shared by each user on a given physical computer. This configuration allows gives each student a "virtual network" on their machine and provides some flexibility and creativity for the instructors and the students. The possible virtual network configurations are almost limitless.

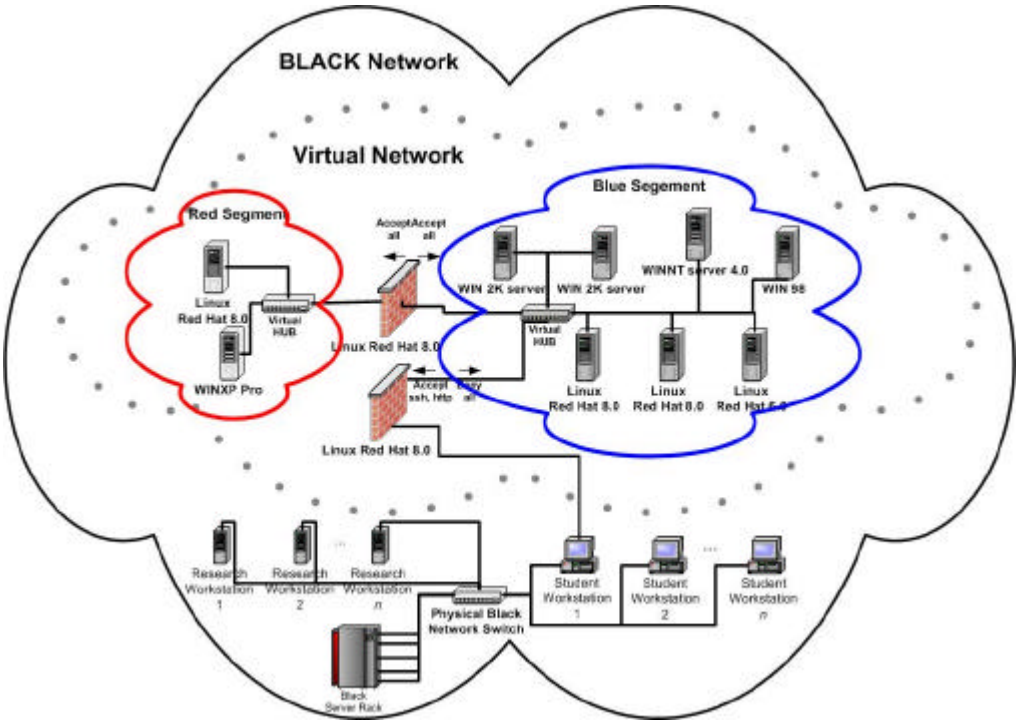

Figure 3: Virtual Information Assurance Network (VIAN)

The VIAN solution offers many advantages over a typical laboratory configuration. They are:

- Adding additional guest systems. The entire configuration file for a guest OS is contained in a folder that, depending on operating system type, ranges from 500 MBtyes to 4.5 GBytes. Additional guest systems or copies of an existing gust OS can be added by simply copying the virtual machine directory.

- The guest operating system is a full and complete installation of the operating system. All system calls and network protocol operations are executed exactly as if the guest operating system was on its own hardware.

- If an exploit or a configuration change causes significant damage to a guest operating system, it can simply be deleted and copied back from an archive location.

- The guest operating system is capable of operating in non-persistent state. This means that when a guest operating system is rebooted, any damage or intentional changes made in the previous session are discarded when the machine ends its session.

- A user is able to exercise complete administrative control over a group of machines and network components. The number of guest operating systems that can exist on a given host is limited only by the amount of hard disk space.

- The ability to operate as an administrator provides an invaluable hands-on learning experience.

- The virtual network can operate in isolation *or* interact with external networks.

We have shown VIAN can provide a highly mobile and robust solution by implementing it on a laptop. We have successfully used this methodology for teaching other courses. We have implemented the virtual network described here on a 1.8 GHz laptop with 1 GByte of RAM and a 30 GByte hard drive. We run 5 and 6 guest operating systems with out any significant latency in any of the guest systems or the host system.

As far as providing an "active learning" environment, the VIAN solution offers the students direct hands-on experience in:

- Scanning and vulnerability testing.
  Port scanning is very bandwidth intensive; 18 students scanning our physical network rapidly reduces bandwidth on the network and possibly place a given machine in a denial of service situation. Each student scanning his own virtual network provides everyone with valuable hands on experience, while eliminating the possibility of network/service failure.

- Hardening multiple operating systems.
  The students also are required to examine the various guest operating systems and go through the steps of hardening them against known exploits and common vulnerabilities. In addition to hardening an existing guest OS, the student could install, configure, and then harden an operating system using a security checklist such as from SANS or the NSA. The virtual machines enable the student to perform these functions without having to worry about tampering with the base operating system's configuration. This task would not be economically or space wise supportable in a physical lab where each OS was installed on a separate system.

- Installing, configuring, and testing security tools.
  In conjunction with scanning and OS hardening (and throughout the remainder of the course) students must install and evaluate a series of security tools. For many of these applications, configuration and use is far from trivial. During the course, the students will use the VIAN to select, deploy, and configure all the tools necessary to establish and defend their network. They go through a formal decision making process where they evaluate and test each tool. The VIAN provides an excellent resource for this process by enabling each student to have a dedicated network with a variety of operating systems to conduct their investigation.

- Seeing exploits in an isolated (and non-persistent if desired) network.
  The traffic in the virtual network is typically very quiet. This enables a student to monitor network traffic and see only packets that relate to the exact sequence of events leading up to a given exploit. For example, a student may exploit an IIS buffer overflow on one of the guest systems and the packet logs and TCP stream information would show only the traffic associated with the scanning and exploit. On most networks the variety of services running will make network *sniffer* logs very difficult to interpret. For an untrained observer (student), this detracts greatly from the learning environment.

- Building Exploits.
  A valuable exercise for students is to actually build an exploit. While this can be effectively done in most common laboratory configurations, it is easier to test the exploit when you have control over the configuration of the operating system and application being exploited.

- Planning general defensive measures.
  From a defensive perspective, we use the virtual machines to demonstrate the concept of firewalls, intrusion detection systems, and forensics analysis. The firewall separating the *red* and *blue* networks is initially set with an "allow all" policy. The student then configures it in an exercise to a "deny all" policy with rules to allow specific traffic such as http and ssh. Using ipchains, a stateless packet-filtering firewall, we can demonstrate the advantages of a packet-filtering firewall. In order to show a stateful packet-filtering firewall we can then use iptables. Finally, in order to take the exercise to the final level, we can install a proxy server such as Squid that filters application (in Squid's case, http) traffic. Similar exercises can be performed with intrusion detection systems.

VIAN, while very well suited for IA education, is also well suited for other computer science courses. We have explored the possibility of using the virtual network for Operating Systems and Networking courses and are currently using our portable solution for two mandatory Information Technology courses. In all circumstances, providing each student with a robust network where they are the network and systems administrator is almost exclusively not supportable in standard computer science laboratory configurations.

Other Components

The IWAR laboratory supports other networks briefly described here. The Cyber Defense network supports the cadets' capstone project in the Information Assurance course, the Cyber Defense exercise. This network is planned, designed, and defended against attacks from the National Security Agency's "red" team entirely by the cadets enrolled in the course. Leveraging the information they learned using the IA network, the cadets apply their knowledge in a "live" exercise. More information on the Cyber Defense exercise can be found in (Welch et al., 2002).

The purpose of the search box network is to provide the ability to find and download offensive and defensive tools. The search box network is a set of four laptops configured for wireless communication through a standard IEEE 802.11b wireless access point. The wireless access point is connected to a cable modem connection through a local ISP. Students and faculty download the tools to a zip drive and then bring the zip disk into the laboratory where they can experiment and use the tool. Policies are in place that require the laptops and zip disks to stay within the confines for the IWAR laboratory.

Related Solutions

Primarily due to the increasing importance of IA education, several colleges and universities are beginning to invest resources towards the construction of information security laboratories (Francia and Smith, 2002, Kaucher and Saunders, 2002, Yasinsac et al., 2002). Others have been looking at using simulation-based tools to educate their students (Carver et al., 2002, Saunders, 2002, VanPutte, 2001). To the best of our knowledge, no one has attempted to design and implement a laboratory on the scale or complexity currently exhibited by the IWAR laboratory. Others have created laboratories, primarily to serve more

general purposes other than specifically information warfare, and few have the similar heterogeneous nature or scale that the IWAR demonstrates.

Kaucher and Saunders describe an Information Assurance laboratory that they use at the National Defense University for educating information assurance and information security professionals. Their network serves a different purpose and thus does not need to be the same scale or complexity as we have built into the IWAR laboratory. Similar characteristics include a heterogeneous network. One of the unique features of their network is that they expose the entire network to their students. This works well for their particular situation, as their students often need to see the entire network to "demystify the technology." (Kaucher and Saunders, 2002) Our VIAN solution "exposes" the network controlled by the student. However, the rest of the networks in the laboratory are not exposed thereby enabling students to explore various reconnaissance techniques from the attackers perspective and then to see, from the defender's perspective, how much information systems expose with default configurations.

Others have taken heterogeneous networking to another level by implementing different layer 2 architectures such as Ethernet, Asynchronous Transfer Mode (ATM), and Fiber Distributed Data Interface (FDDI) on a token ring (Francia and Smith, 2002). Their network design is different from ours in that they are using it more for system modelling and simulation, networking, and special projects rather than information warfare. The scope of their network is also much smaller and where our heterogeneous nature consists of multiple operating systems and services, their heterogeneous flavour is a result of different link layer protocols. Some similarities also exist, however. We have established a wireless network using the IEEE 802.11b protocol as an alternate data link layer protocol liable to different types of exploits than what one would see on an Ethernet network.

Yasinsac describes a computer security laboratory project for outreach, research, and education. Their laboratory serves a similar purpose as the IWAR laboratory but on a smaller scale. Similar to the IWAR laboratory, they have been challenged to provide an environment where students are free to explore without creating administratively challenging headaches when systems break because of the use of certain tools. Similar to our VIAN solution, their laboratory uses VMware (Yasinsac et al., 2002) to control computer configuration and contain exploits. Our usage of VMware is more so to provide a virtual network of systems that the students have administrative control. Because of the isolated nature of our networks, we give the students full administrative control over their virtual network and their host machine. There are a few machines that are off limits because they contain important services (i.e. the main file server); however, they are backed up routinely in case of an exploit that compromises the machine.

Others have begun designing or looking at simulation based-tools to educate others in IA. However, to the best of our knowledge, many tools exist that model networks, but no tool exists that accurately models the specific decisions that must be made to simulate an IA education. (Carver et al., 2002) The implementation and maintenance of an IWAR type laboratory requires significant investments in terms of hardware, software, and human resources to build and maintain the physical networks of computers and communication components. This is not a unique problem. We agree that a tool or model that can be used by students to assess the quality of their information system design choices prior to (or instead of) a physical implementation is required in an IA education. As is true in military training

exercises however, simulation-based tools will always complement, rather than replace a hands-on "live-fire" experience.

Results

The design, implementation, and maintenance of the IWAR laboratory required an extraordinaire amount of effort from the faculty, support staff, and students. It is the centrepiece of the Information Assurance program at USMA. In addition to the hands-on, active learning capability it provides for teaching students information warfare topics, several other positive outcomes have been observed that can be directly contributed to the laboratory.

The IWAR laboratory is touching all facets of IA education at West Point. As a result of the VIAN solution being ported to laptops, all cadets at USMA will experience, hands-on experience with offensive and defensive information operational tools. As of the upcoming school year, all cadets must take two Information Technology courses. One eighth of their lessons will focus on information assurance topics. The VIAN solution is how they will receive their hands-on, active-learning education. For the third year, a course on the "Policy and Strategy of CyberWarfare" is being taught to political science majors with the IWAR laboratory being used for half of their lessons. In part because of the opportunity to use the SIGSAC network, the membership of the SIGSAC student club has increased to approximately 10% of the student population at USMA. Finally, there has been a large, noticeable increase in the number of IA related summer internships that the cadets participate in for both Department of Defense and commercial organizations.

As awareness in Information Assurance has increased throughout the student population, the faculty has had to modify course content to include more in-depth to the instruction. This was evidence during this year's cyber defense exercise as the cadets built a complicated and formidable defense that others are using as positive examples in their instruction to real world network and system administrators. As a result the faculty has had to increase their own knowledge in the subject matter. The IWAR laboratory provides faculty with a facility to learn about emerging information warfare topics.

As USMA attracts many visitors from the upper levels of the federal government and Department of Defense, the IWAR laboratory has been a focal point of interest for such visitors. Not only does it expose the outstanding work by both students and faculty, but a reciprocal effect is taking place, where after a demonstration of the capabilities of both offensive and defensive techniques, the visitors' leave understanding in much clearer terms what the threats to and the vulnerabilities of information systems are. In some cases, they bring back their lessons learned to their organizations and influence decision makers at all levels to pay heed to prioritizing efforts in this area.

Finally, the IWAR laboratory has had a direct, and sometimes immediate, influence on our graduating students. Recently an article about the Cyber Defense exercise was published in Government Computer News. One of the individuals interviewed was a former student, now a First Lieutenant in the United States Army. He was a member of the first West Point class to be exposed to the original IWAR laboratory. In the article the former student was quoted as stating:

> "I've been putting what I learned into practice in a real environment for several months now. I am writing this from the V Corps main HQ at one of Saddam Hussein's presidential palace compounds in Baghdad. The lessons

> learned in the exercise have come into play here. I can't go into too much detail about the network, but it has 300-plus routers and is still growing. It is the largest tactical network ever." (Jackson, 2003)

Such an example highlight that the experiences learned in the IWAR laboratory directly translate to solutions in the real world. The IWAR laboratory component of the IA educational program at West Point provides a much richer experience for students than what classroom instruction alone could provide.

Future Work

The key to implementing future work is to make incremental changes and learn from their lessons. We plan to follow this approach as we continue to refine and improve the laboratory so that we continue to provide a quality IA education to our students and sustained research work for our Army.

We plan on packaging the VIAN solution along with modular lessons and exercises for use at other institutions. We realize that not every organization may have the resources to build and maintain a laboratory at the same scope of an IWAR laboratory. However, we believe any institution, could take the VIAN solution and use it too successfully teach IA topics in an active learning environment. We have been collaborating with several courses within USMA and other institutions outside to determine the details of such a solution.

The second priority to improve the usability of the laboratory for both students and faculty is to categorize the attack tools in such a way that the student could "pull the tool off the shelf", read the instructions and within a sufficient amount of time be able to see the attack. Because exploits are very specifically targeted against an operating system/application pair, having such a capability would significantly increase the quality of instruction and the time allocated for preparing classes and demonstrations.

Conclusion

We have presented our work in creating an isolated network whereby students and faculty can build, configure, maintain their own systems and employ both offensive and defensive cyber warfare tools. Without such a hands-on experience, teaching technical topics in Information Warfare is futile. The justification for the laboratory is clear—in order to provide a quality education for our students and an environment where relevant applied research can be conducted, the hands-on experiences acquired using the laboratory's networks cannot be replaced by PowerPoint presentations or simulations. Isolation of the network is a must in order to avoid leakage of exploits or self-induced denial of service attacks onto a live, production network. The development and improvement of the laboratory over time has proved to deliver sustainable results such as stimulating interest, increasing general awareness, improving research quality, and delivering graduates to the public with a much stronger background in information warfare and information assurance.

References

Carver, C. A. J., Surdu, J. R., Hill, J. M. D., Ragsdale , D. J., Lathrop, S. D. and Presby, T. (2002) In *3rd Annual IEEE Information Assurance Workshop*West Point, NY.

Department of Defense Report to Congress *Network Centric Warfare*, [On-line] http://www.c3i.osd.mil/NCW/, [Accessed on 23 April 2002].

Felder, R. M. (1993) Reaching the Second Tie--Learning and Teaching Styles in College Science Education. *Journal of College Science Teaching,* **23**:286-290.

Francia, G. A. and Smith, R. K. (2002) The Design and Implementation of a Heterogeneous Computer Networking Laboratory. *The Journal of Information Security,* **1**(2).

Garamone, J. (2001) *Capabilities, Strategy Must Converge to Face New Threats,* [On-line] http://www.defenselink.mil/news/Jun2001/n06222001_200106221.html, [Accessed on 9 May 2002].

Jackson, W. 2003 'Cyberdrill carries over to real war' *Government Computer News,* **22**, 11.

Kaucher, C. E. and Saunders, J. H. (2002) Building an information assurance laboratory for graduate-level education. *The Journal of Information Security,* **1**(2).

Moore, D., Paxson, V., Savage, S., Shannon, C., Staniford, S. and Weaver, N. *The Spread of the Sapphire/Slammer Worm*,
[On-line] http://www.cs.berkeley.edu/~nweaver/sapphire/, [Accessed on 31 May 2003].

Saunders, J. H. (2002) Simulation Approachs in Information Security Education. *The Journal of Information Security,* **1**(2).

VanPutte, M. (2001) *SimSecurity - Distance Learning and Virtual Laboratory for Information Assurance,* [On-line]
http://www.movesinstitute.org/OpenHouse2001/Presentations/VanPutteSimSecurity.ppt,
[Accessed on 19 December 2002].

VMWare (2001) *VMware Workstation 3.0 User's Manual,* VMware, Inc., Palo Alto, CA

Welch, D. W., Ragsdale, D. J., Schepens, W. (2002) Trial by Fire in Information Assurance Education, *IEEE Computer*, April 2002, pp. 30-37.

Yasinsac, A., Frazier, J., Bogdanov, M. (2002) Developing an Academic Security Laboratory**.** *The Journal of Information Security,* 1(2**).**

Design and Implementation of an Information Security Laboratory

V. Padman, N.Memon, P. Frankl and G. Naumovich

Department of Computer Science
Polytechnic University, U.S.A
E-mail: vikram@computer.org, {memon, frankl, naumovich}@poly.edu

Abstract

It has been recognized for some time now that education in information security is better served by a laboratory component that reinforces principle and theoretical analysis learnt in the class room with a follow-up hands-on component performed in an appropriate laboratory. In this paper we present the design of a highly reconfigurable laboratory for information security education. The design has been implemented successfully in ISIS - The Information Systems and Internet Security Laboratory at Polytechnic University. We also describe the rationale for our design and give examples of a few typical assignments that the laboratory facilitates.

Introduction

The recent focus on security education, kindled by the NSA Center of Excellence in Education program (NSA 2002) has seen a variety of universities add a security component to their computer science and engineering curriculum. As a result, we now have 36 universities that have been designated as Centers of Excellence in education. However, a significant number of programs continue to teach information security in the decades old traditional framework, focusing solely on theoretical principles and their analysis. Although theoretical concepts are essential and need to be taught, it is very important to also show students how to apply the theory they have learnt in very different and important practical situations. Hence, a good part of an information security course should also focus on applications and operational concerns. In order to do this, a supporting laboratory becomes necessary.

Recent years have seen an increased awareness on the importance of a laboratory component in information security education (Irvine 1999, Bishop 1999, Hill et al. 2001, Mateti 2003). Irvine (1999) points out that securing a system requires a "marriage" of good science and engineering. And that engineering components are best taught by reinforcing concepts taught in the class by hands-on experiences in the laboratory. She further points out that just as it is unreasonable to expect a student to learn programming only by reading about it, it is also unreasonable to expect students to learn "security engineering" solely from discussions in the class room. Similarly, Hill et al. (2001) and Mateti (2003) also make the case for laboratory based instruction in information security and in fact provide detailed examples of specific courses and lab projects that accomplish this goal.

A laboratory for information security education can be designed in a different manner depending on the nature of the program and the course being serviced. However, there are certain general principles that guide the design of such a laboratory. Specifically, a well designed laboratory should possess the following characteristics:

ISSN 1445-3312 print/ISSN 1445-3347 online

Reconfigurable: The lab should be highly flexible and re-configurable. Different topics and assignments require different operating systems and/or network topologies and it should be possible to change the configuration of hosts and networks easily and efficiently.

Heterogeneous: The lab should comprise of multiple platforms from multiple vendors. A lab with homogeneous environment does not effectively train students to cope with real world situations.

Scalable: The lab should be scalable and should be able to sustain many students, and still have enough duties for each student to handle. Student groups should not get large due to lack of resources.

Cost Effective: The cost of setup and maintenance of the lab must be far less them what's being simulated by the lab. For example, the lab should effectively simulate a small to medium enterprise network but the cost for building and maintaining the lab should be far less then the cost of a moderate enterprise network.

Robust: The lab should be able to sustain and handle inadvertent damage by the students. For example, it should be possible to quickly recover the set-up and configuration of a host node even after a student accidentally causes a malicious program to erase the hard disk.

Maintainable: The lab should be easy to maintain. Routine tasks like back-up and application of software patches should be easy to perform and automated to whatever degree possible.

Realistic: The lab should provide practical and first hand experience to students in a network environment that is close, in terms of complexity, to a network that they might encounter in a real world enterprise.

Insulated: Activities in the lab should not affect traffic on the campus network. There should be sufficient amount of separation and isolation enforced between the lab network and the external network. The presence of the lab should not be a cause of concern to campus network authorities.

In this paper we describe the design of ISIS - An Information Systems and Internet Security Laboratory at Polytechnic University, which aims to achieve the above listed design goals. ISIS was initially started as the result of an NSF CCLI grant to develop a sequence of undergraduate courses in computer and network security and an accompanying laboratory. Initial lab and course design was done with the assistance of Information Systems Security Laboratory (ISSL 2001) at Iowa State University which has long been an NSA designated Center of Academic Excellence in information assurance education and research. ISIS has been running for more than two years now and the lab and the courses it supports have proved to be immensely successful. In fact the role of ISIS has been significantly expanded beyond its original scope and design and it now serves as a center of education and research in information assurance at Polytechnic University.

The rest of this paper is organized as follows: In the next section we describe the overall architecture of ISIS and two of its smaller components - the student workstation network and

the server cluster. In ASSET- “A Secure Systems Experimentation Test bed” section we describe in detail the design of the core of ISIS. In ‘Example Class Assignment’ section we describe briefly some typical assignments supported by ISIS and we conclude with a brief discussion on future plans for expanding ISIS.

ISIS Architecture

ISIS consists of heterogeneous platforms and multiple interconnected networks to facilitate hands-on experimentation and project work in issues related to information security. ISIS lab is divided physically and logically into four areas, namely:

i. The Student Workstation Network,
ii. The Server Cluster,
iii. A Secure Systems Experimentation Test bed (ASSET).
iv. A VPN Concentrator

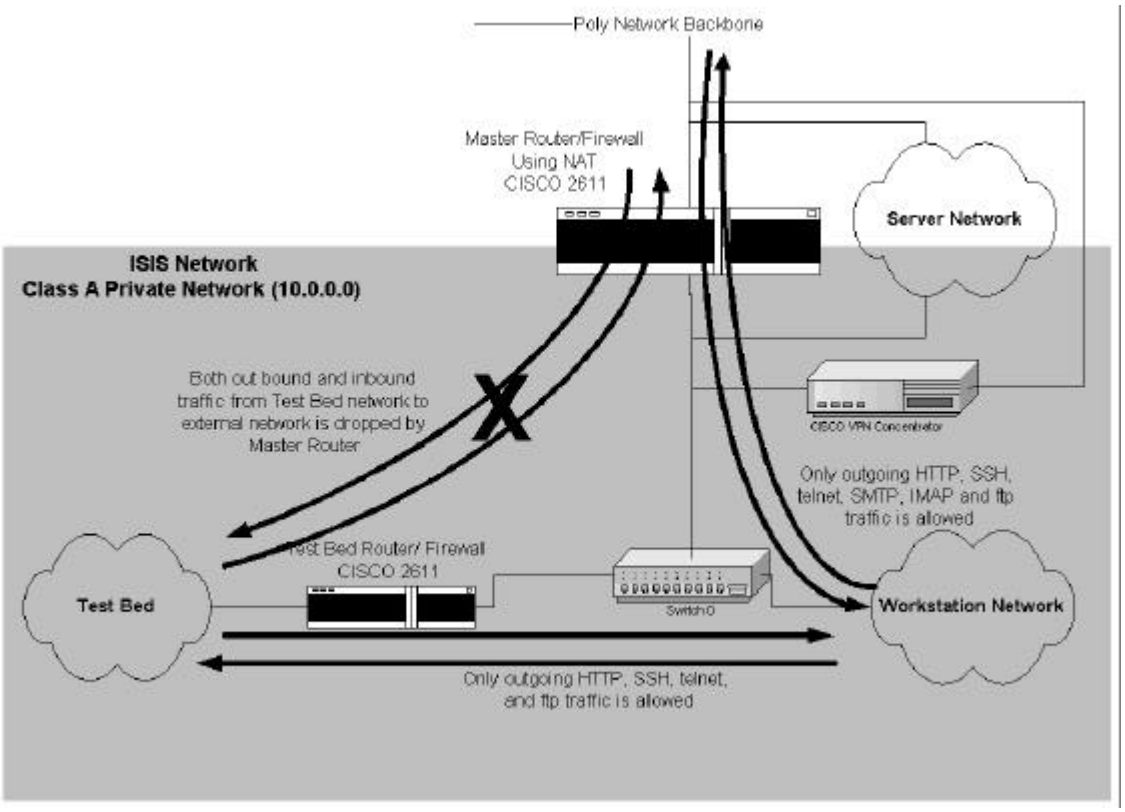

Figure 1: ISIS lab architecture overview showing its three main components - 1) ASSET - A Secure Systems Experimentation TestBed, 2) The Student Workstation Network, and 3) The Server Cluster - and their interconnection with each other and with the campus network backbone

The above figure shows how these four components are interconnected with each other and also with the external campus network. The student workstation network and the test bed ASSET are inside a class A private network so that they are isolated from traffic on the campus network and the internet. The private network is created using a router with NAT capabilities. This router is shown in Figure 1 labelled as ``Master Router/Firewall''. Usually a private network is created to hide internal network topology and expand the range of available IP address. In our case it is critical to separate our network traffic from external

network in order to stop internal traffic, malicious and otherwise, from reaching the external network. The router will prevent packets from internal traffic to escape out into the external network.

The second advantage that a Class A private network provides is the large number of subnets that can be created within it. We could potentially have 2^{16} subnets with 250 hosts in each subnet in our network. This fact is critical to the design of our test bed network, as is explained in the 'ASSET- A Secure System Experimentation TestBed' section.

In addition to performing NAT, the master router is also configured to act as a firewall to impose restriction on traffic flowing to and from the internal network. Furthermore, traffic from and to the test bed network from the workstation network and the server network is restricted by a second firewall labelled as the Test bed/Router Firewall in Figure 1. This ensures that any attack traffic in the test bed network does not enter the workstation network or the server network.

The VPN concentrator is used to provide connectivity to ISIS network from remote locations using a private VPN tunnel.

The Student Workstation Network

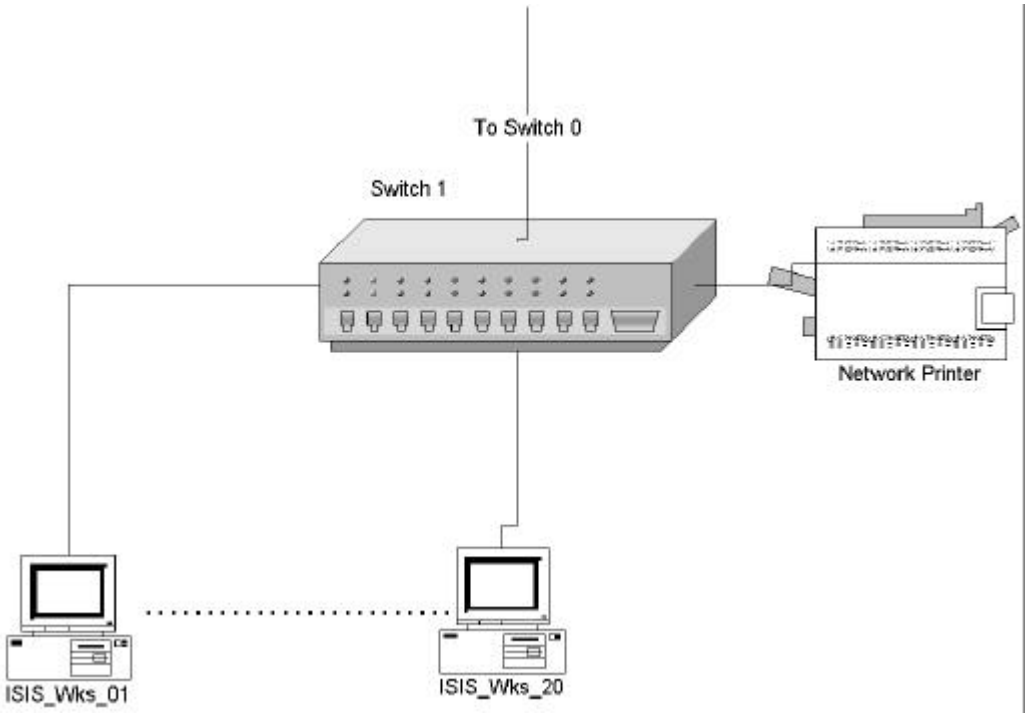

Figure 2: Physical network layout of the student workstation network.

The primary purpose of the workstation network is to provide students a means to access the ASSET network. Typically, for most assignments, students have to be physically present in the lab and logged on to a workstation in order to access ASSET. The workstations themselves are Pentium 4, 1.5 GHz general-purpose machines, running Windows 2000 and equipped with standard university lab software, like compilers, editors etc. These workstations are members of the ISIS active directory server present in the server network. Currently there are 20 workstations as shown in Figure 2.

Individual workstations in this network are completely locked down physically by using padlocks on the machines and also by appropriate configuration of BIOS settings and Windows domain policies and restrictions. For example, students cannot reboot these machine using bootable floppies or CD's. They cannot install or remove any software, and connect or disconnect these computers from the network. They are only allowed to use the applications that are installed on the workstations. The software restrictions are enforced using Windows 2000 domain policies and restriction suggested by NSA's Windows 2000 lock down guidelines (http://www.nsa.gov/snac/index.html).

Although students are only allowed to store their files on the file server and not in the workstation they work on, they are still provided some writable space in each workstation. Without user writable space, Windows will not allow any user to log on. This space is very small (10MB) and each workstation is cleaned occasionally by erasing all users temporary directories, and/or re-installing a fresh image, if necessary, during the cleaning process.

The Server Cluster

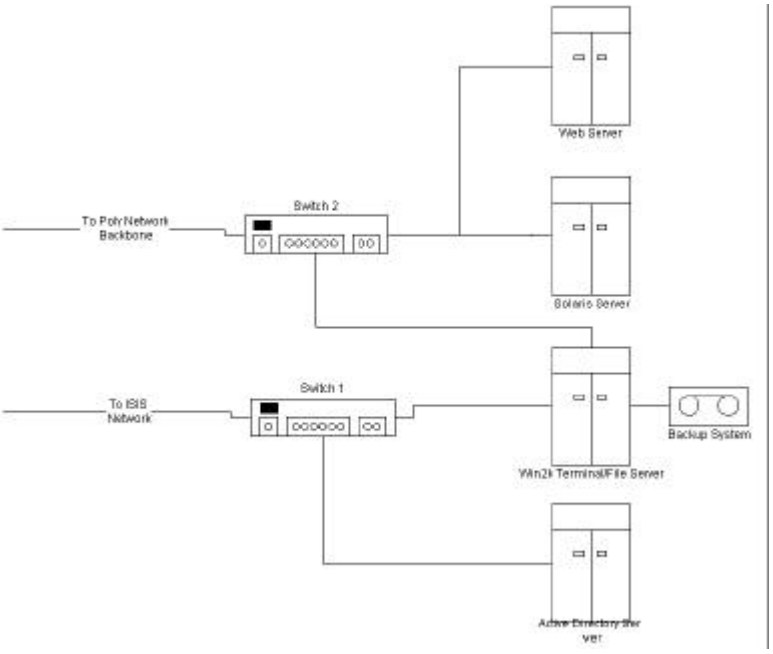

Figure 3: Physical network layout of the server network.

The server component of ISIS currently is composed of four serves: 1) A Web server, 2) A Solaris server, 3) A Win2k Terminal/File server and 4) An Active Directory server. The web server is used to host lab's and students web pages. The Solaris and Win2k terminal servers are used by the students for compute intensive tasks like password cracking and cryptanalysis. These servers also contain a repository of security related tools that students need for their projects and assignments. The active directory server is used to manage the ISIS lab active directory. The Win2k server is also used as a file server to store student files. Each student is allowed to store up to 5GB's in the file server and their files are automatically backed up by the backup system and also screened for common viruses frequently. The total storage capacity on the server network exceeds half a terabyte.

The server network can facilitate secure remote access to our network via the Windows terminal server. We use a dual homed Win2k terminal server for remote access. The remote access server is also part of the ISIS active directory; so all users in the active directory can potentially access the secure systems experimental test bed from a remote location. This was done to facilitate students who cannot be physically present in the lab.

ASSET- A Secure Systems Experimentation TestBed

In this section we describe the third component of ISIS - A Secure Systems Experimentation Test bed (in short ASSET). ASSET is the core of the lab and this is where most of the lab activities take place. It consists of a highly reconfigurable network built around 2 layer 2 switch, 32 computers fitted with two or more NICs and removable hard drives, two VMware ESX servers, 16 CISCO 2611 routers, and 1 CISCO 4235 IDS sensor as shown in Figure 4 and 5.

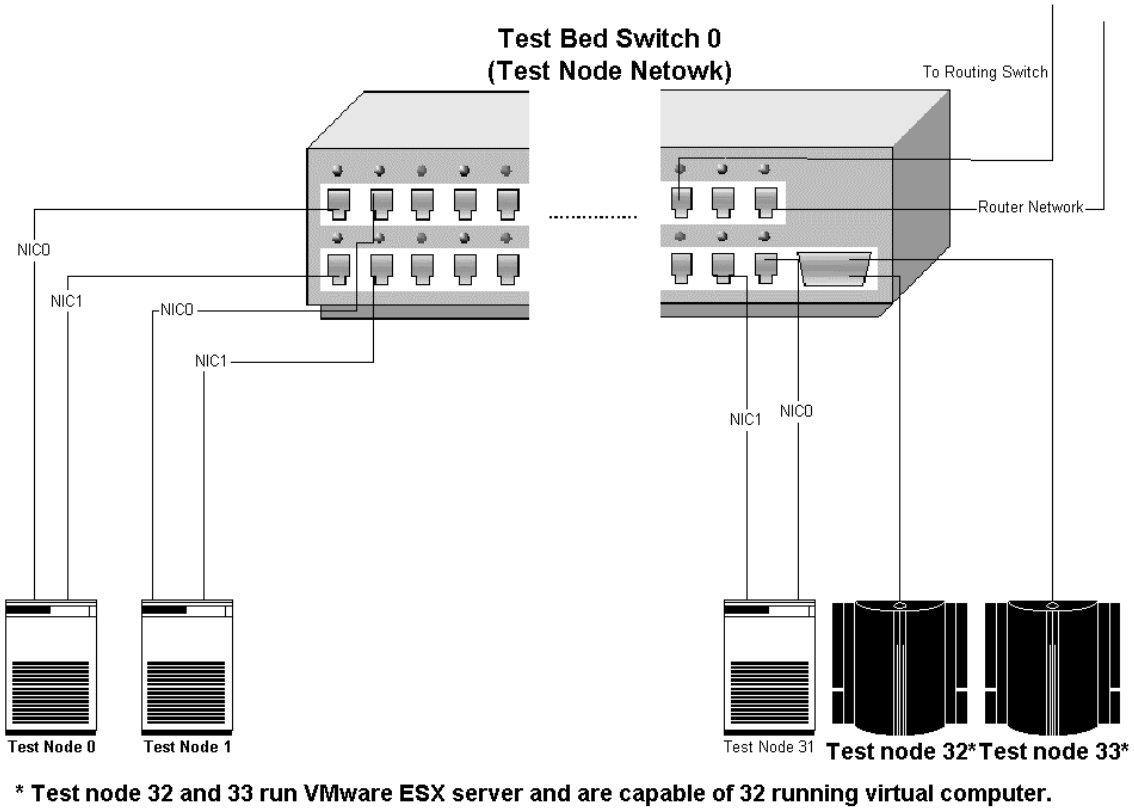

Figure 4: Physical layout of ASSET's test node network.

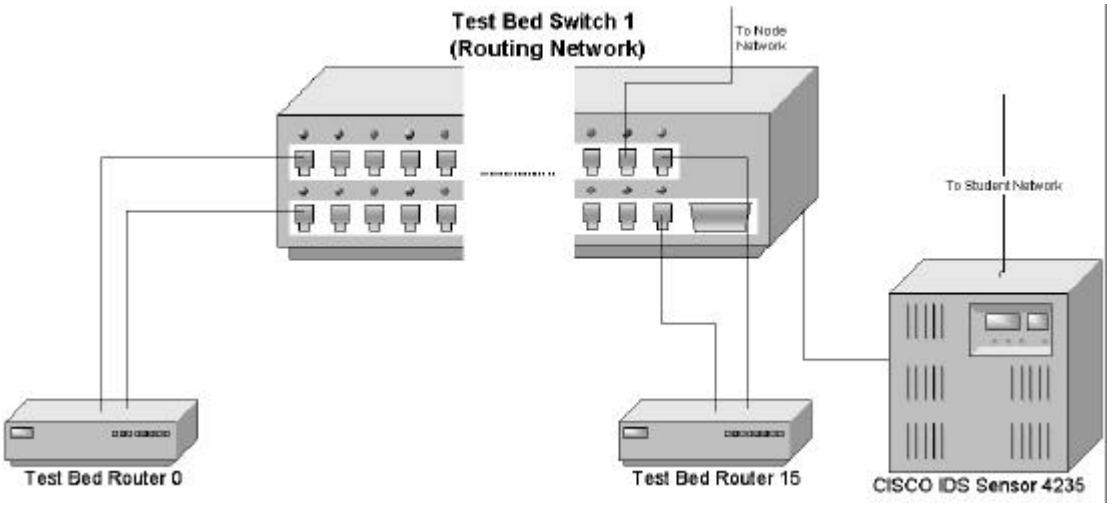

Figure 1: Physical layout of ASSET's router network

ASSET is designed to be flexible and highly reconfigurable. Flexibility in terms of network layout and software running on end nodes is necessary as this allows us to support assignments of vastly different nature and scope with the same resources. The need for flexibility in terms of operating system can be seen in assignments like Linux and Windows hardening.

These assignments follow each other and it's important that we are able to change the operating system on a large number of nodes in a reasonable amount of time (say a few hours). A dual boot architecture will not suffice, as we need the ability to load ASSET hosts with selected versions of an operating system with deliberate misconfigurations for students to discover and fix. It is also important that we have the ability to restore these nodes to a default state in minutes, after inadvertent damage by a student. In order to achieve these goals we use nodes with removable hard drives. This way we can load or restore a node by simple replacing its hard drive. A hard drive duplicator allows us to load the same configuration in multiple hosts. Disk images of different versions of different operating systems, and applications with and without flaws, are stored on the server network and can be copied on to a disk in minutes, and can then be duplicated and loaded in multiple hosts.

The need for flexibility in terms of network layout arises from the variety of network security assignments that students are required to perform, ranging from network footprinting, penetration testing, intrusion detection and prevention, and finally war games. Again, such assignments follow each other in a logical sequence and one needs the ability to reconfigure the network topology from one assignment to the other in order to meet the specification of each assignment. Furthermore, in certain more complex assignments it is also desirable to mimic a slowly changing enterprise network and this leads to the need for an ability to automatically change network layout by means of scripts and without human intervention.

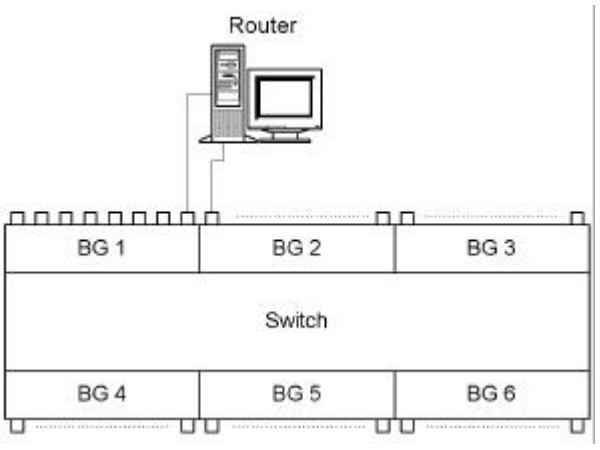

Figure 6: A switch divided into 6 bridge groups, each bridge group represents a VLAN. Figure also shows how two bridge groups could be connected using a router

To make ASSET flexible in terms of network layout we use Virtual LAN's (VLAN's) and create logical networks. We do this using a switch with VLAN support. In a conventional switch, all ports belong to the same broadcast domain (i.e. one switch represents a network segment) and many networks can be created using multiple switches interconnected with a router to represent an enterprise network. An example of such a network is depicted in Figure 7. With a switch that can support multiple VLAN's, it is possible to create such a network without having multiple switches, and we can change the network layout without changing the physical layout.

In a switch with VLAN support, VLAN's can be created and modified by changing the software configuration of the switch to which all the hosts on the network are physically connected to. Using VLAN's it is possible to create independent virtual broadcast domains within a switch as shown in Figure 6. Also, a switch with VLAN support can have multiple broadcast domains. We could interconnect these domains by having a router between them (Figure 6) and the network in Figure 7 could be created by configuring the switch to have multiple VLAN's and physically it would look like what is shown in Figure 8.

All routing is done by the routing network except for routing within a group's network. The routing network consist of 16 CISCO 2611 routers, all connected to a second switch, which is called "Test Bed Switch 1". The routing network is only responsible for routing traffic between groups and student network. Since the routers are maintained by a lab monitor, it provides a stable network which is immune to attacks and mistakes done by students. Each students group is assigned to a network and they are responsible for routing within their network, which is normally done by a Linux router built on top of a test node.

The IDS sensor in the routing network is used for monitoring the ASSET network. It is also used in assignments, where its logs are given to students for analysis.

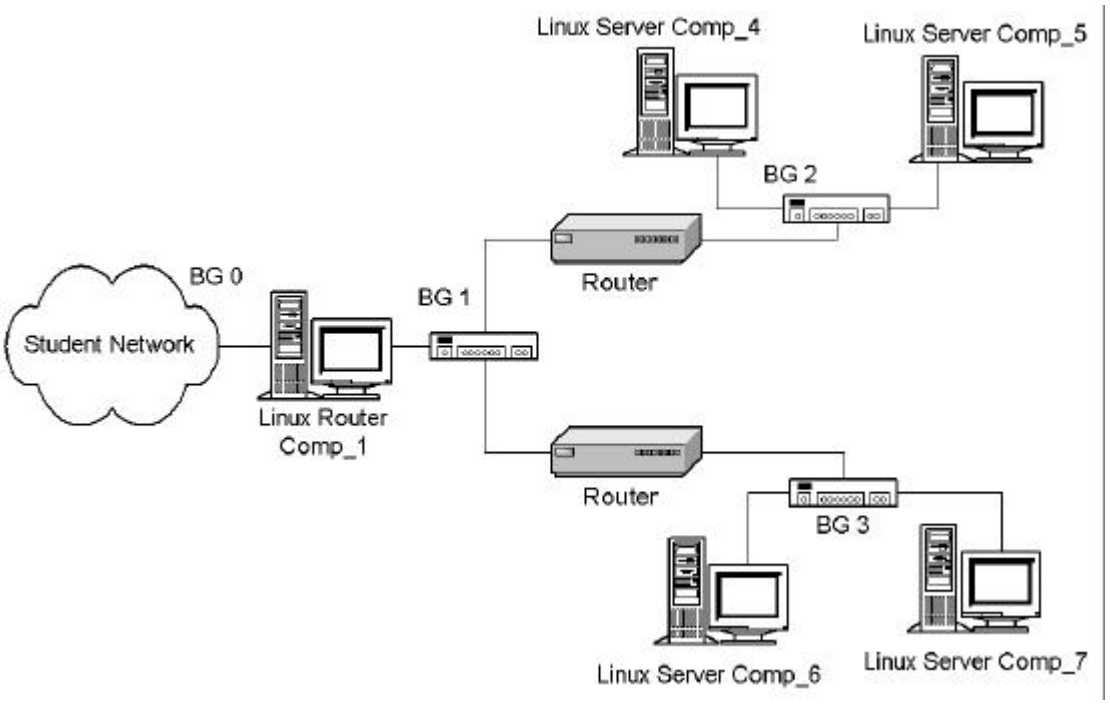

Figure 7: An example of small network with 4 network segments, built using an independent switch to support each network segment. Each network segment is interconnected using a Linux router.

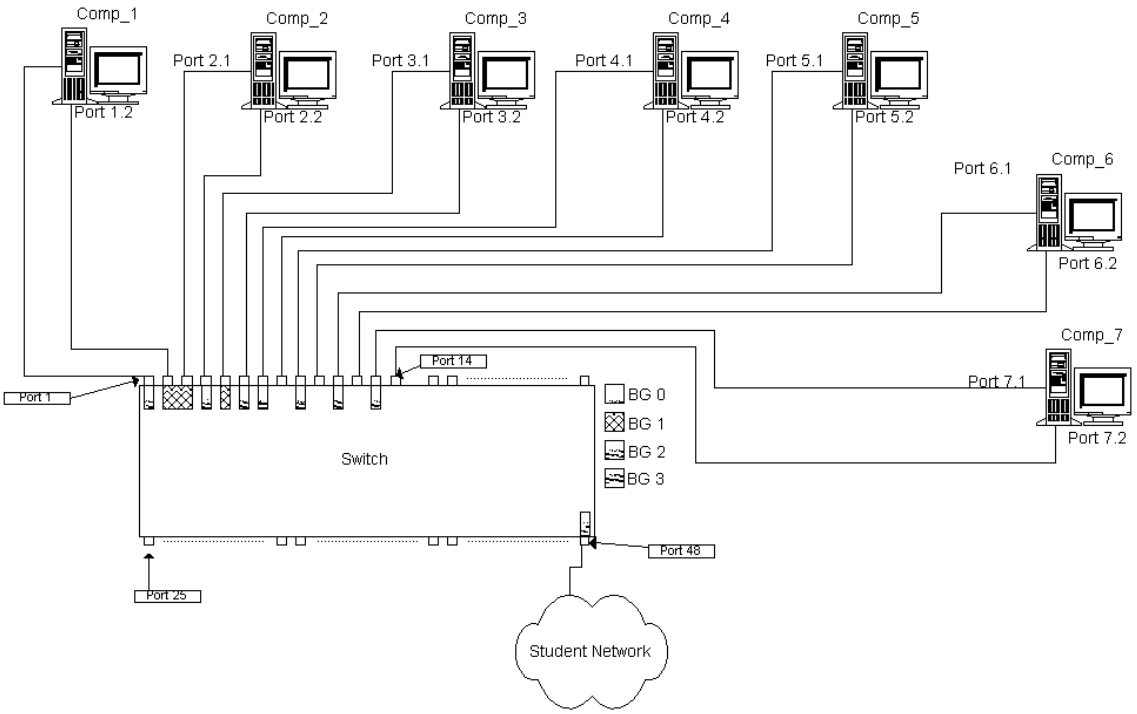

Figure 8: The network depicted in Figure 6 implemented in a single switch configured to support 4 bridge groups.

More complicated networks as shown in Figure 9 can be created using test nodes 32 and 33 in combination with the other 32 computers. These two machines can run up 32 virtual computers using VMware's ESX operating system. ESX server is a mainframe class-computing environment, and is capable of having internal networks independent of the external network. Node 32 and 33 can be used to simulate a changing network. There are 16 virtual test networks inside each node and the network topology is dynamic. Using a skilfully

crafted script one can periodically change the internal network configuration of the VMware network to simulate a changing network. This kind of a network, for example, could be used in routing vulnerability analysis assignments.

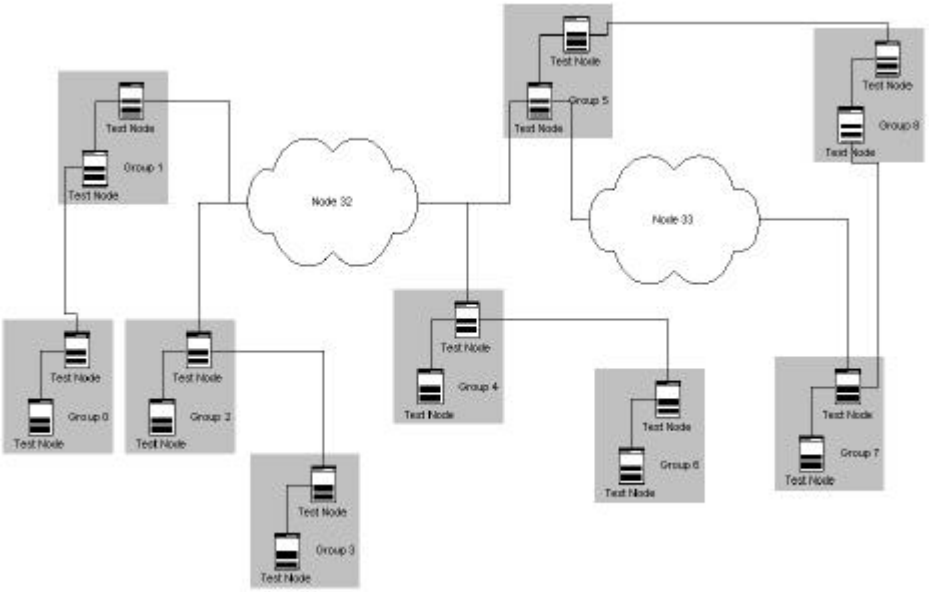

Figure 9: An example logical network that can be built in ISIS testbed network. This example can support 8 student groups. Node 32 and 33 simulate a changing network

From the above discussion, we can see that ASSET meets the design goals of reconfigurability, heterogeneity, maintainability, and robustness. The dual firewalls and private network provide sufficient isolation to yield an insulated environment for experimentation. Since the entire lab can be constructed from cheaply available hardware and open source software, the design is also cost effective. The design is scalable as it allows us to have several networks, thereby facilitating smaller groups even with a class size of 30 or 40 students. Finally, the design allows us to configure different realistic environments for experimentation and exploration.

Example Class Assignments

The laboratory design that we have described facilitates a rich variety of class assignments. A detailed description of the different assignments we have created and their objectives are beyond the scope of this paper. However, in order to describe what the laboratory makes possible, we give a few brief examples below.

Server Assignments

These assignments only utilize one or more of the servers in the server network. Often they are computer intensive in nature. An example of this category of assignments is the following:

- *Investigation of Block Ciphers.* The core learning objectives of this assignment include:

1. Understand the implementation of a modern cryptosystem like AES by examining public domain source code.
2. Understand the difference in security and error propagation properties of the different modes in which a block cipher can be used.

3. Understand how the difficulty of a brute force attack grows exponentially with the length of the encryption key.
4. Understand the confusion and diffusion properties of a modern cryptosystem like DES or AES and compare it with a simple classical cipher.

A similar assignment to the above can also be done with a public key cryptosystem. Other examples include integration of various cryptographic primitives like encryption, authentication and key management to create a file encryption and authentication utility; designing and implementing a simple virus scanning utility; and understanding and using some standard password cracking tools.

Host Assignments

Here the ASSET network is configured as a flat network of hosts and each is assigned to a host. Assignments for this type of configuration typically explore security of stand-alone computer system. An example is the following:

- *Setting up, exploiting and detecting a Trojan horse.* This assignment is divided into three parts. In the first part all nodes are assigned a known root password. Students are asked to install Trojan horses in these machines which will allow them access to some resource in the machine even if the root password is changed. In Part II, each student is randomly assigned to one of the compromised nodes from the first part and they are asked to detect and remove all Trojans they can find. Finally, in Part III, the students are asked to exploit other nodes by making use of the back doors they have planted.

Other examples include hardening a poorly configured Windows and/or Linux machine as per security guidelines specified by the NSA. Assignments involving malicious code are also performed with such a configuration. For example, learning about robust programming techniques in general and designing and exploiting buffer overflows and format string vulnerabilities in particular.

Network Assignments

These assignments require configuration of the ASSET into a collection of networks or clouds of networks and student tasks include exploring, configuring, and defending a network. Example assignments include:

1. Exploiting and understanding ARP vulnerabilities, such as ARP cache poisoning and denial of service attacks that can be done through ARP in the local subnet.
2. TCP and UDP vulnerabilities such as session hijacking, spoofing, and other DOS attacks in TCP and UDP.
3. Vulnerabilities in routing protocols such as RIP, and OSPF.
4. Use of network mapping utilities.
5. Secure communication using IPSEC, SSL, and other upper layer protocols.
6. Implementation of secure echo and secure HTTP.
7. Configuring firewalls using IP chains.
8. Assignments involving intrusion detection and prevention.
9. War game like assignments where students attack networks being administered by other students while at the same time defending their own network from attack.

Conclusions and Future Development Plan

In this paper we have described the design of a laboratory for information security education. We argued that the design goals of such a laboratory should include reconfigurability, scalability, robustness, maintainability, cost effectiveness, and heterogeneity. Furthermore the lab should be well insulated from the external network and should provide a realistic environment for student experimentation and learning. The design has been implemented successfully in ISIS - The Information Systems and Internet Security Laboratory at Polytechnic University.

Future plans for expanding ISIS include the addition of a wireless subnet, software and hardware for advanced intrusion detection and prevention and finally equipment that will facilitate lab work in biometrics and computer and network forensics.

References

Bishop, M. (1999) What Do We Mean by Computer Security Education, *22nd National Information Systems Security Conference*, Arlington, Virginia, p 604

Hill, J.M.D., Carver, C.A.Jr., Humphries, J.W., and Pooch, U.W. (2001) Using an Isolated Network Laboratory to Teach Advanced Networks and Security. *Proceedings of ACM SIGCSE Technical Symposium on Computer Science Education*, Charlotte, North Carolina, pp 36-40

Mateti, P. (2003) A Laboratory-Based Course on Internet Security, *Proceedings of ACM SIGCSE Technical Symposium on Computer Science Education*, Reno, Nevada.

Irvine, C.E. (1999) *Amplifying* Security Education in the Laboratory. *Proceedings IFIP TC11 WC 11.8 First World Conference on Information Security Education*, pp 139 146, Kista, Sweden.

ISSL (2001) *Information Systems Security Laboratory, Iowa State University.* [Internet] Available from: <http://www.issl.org/ [Accessed December, 2001]

NSA (2002) *Centers of Academic Excellence in Information Assurance Education.* [Internet] Available from: <http://www.nsa.gov/isso/programs/coeiae/index.htm [Accessed August, 2002]

Teaching Coursework Master – Computer Forensics

C. J. Armstrong

School of Information Systems
Curtin University, Western Australia.
E-mail: armstrongc@cbs.curtin.edu;

Abstract

This paper discusses aspects of the computer forensic course taught in a Masters degree at Curtin University, Western Australia. Computer forensics is important to both the business and law enforcement environments. Along the path from the enactment of a crime through to the due process of courtroom litigation there are numerous obstacles that may challenge law enforcement and or security practitioners. This paper discusses teaching computer forensics as one of the roles that academia may play in assisting security practitioners and law enforcement agencies investigating computer related crime. It recognises industry requirements and addresses an academic response for the need for computer forensic training and education by providing an overview of the computer forensic course taught in a Masters degree at Curtin University.

Introduction

Police are responsible for upholding the law and investigating, apprehending and prosecuting breaches of the law. The successful prosecution of computer based crime is reliant upon the investigator being able to prove beyond a reasonable doubt who, what, how and when a criminal event occurred within the stringent principles of forensic examination of evidence. Computer crime is of such a nature that it is often difficult for the general public to perceive or to understand that a crime has actually occurred. Criminals are using computers to store records regarding drug deals, money laundering, embezzlement, mail fraud, telemarketing fraud, prostitution, gambling matters, extortion, and a myriad of other criminal activities (Icove et al, 1995). The victim may be a large corporation, may be far away, or may be considered an unfriendly nation, competitor or even an enemy.

The nature of the Internet provides a borderless environment, easy anonymity, concealment of activities and new low cost tools with which to perpetrate crime (Vatis, 2000). Evidence that computer and Internet crime incidences continue to in increase is confirmed in publications such as the annual Computer Security Institute report. (Powers, 2002) Computer crime is on the increase but there are indications that some public perception of malicious abuse may be inflated. Furnell, (2002) discusses how some figures relate to reported incidents from one particular set of surveys and that the true level of computer crime may be much higher because much is not reported due to risk of undesirable consequences; bad publicity, legal liability, loss of custom. Also financial loss is only one type of impact. Others include; disruption of service, loss of data, damage of reputation and these are difficult to quantify and could be more significant (Furnell, 2002).

An investigation of computer crime may use tools, procedures and methods not readily be available to the public and therefore not be readily understood and accepted. For these investigative findings to be accepted they must be recognised by other experts within the field

ISSN 1445-3312 print/ISSN 1445-3347 online

and conform to national and international standards of practice. The risks facing a computer forensic investigator include loss of credibility if another expert witness can demonstrate that proper or appropriate courses of action were mismanaged. It is the role of the independent expert to explain technical issues in layman's terms so that the judge, jury, accused, barrister and solicitor alike can understand the evidence put before them (Armstrong, 2002).

This paper discusses the needs of law enforcement and IT professional in computer forensic training and education. An overview of a course in Computer Forensics in a Masters program at Curtin University is also presented.

The Crime to Court Path

Kruse and Heiser (2002) suggest that cyber lawyers are constantly facing a changing legal environment and need to be flexible and learning continually. There are two particular participants in the battle against computer crime that would benefit from education and training in computer forensics. They are the systems administrators of corporate computer systems and the law enforcement investigators. Both may be required to present prosecution evidence in a court of law either as an expert witness or a prosecuting officer of the law. If the representatives for the defence can reduce the credibility of the prosecution case the prosecution may fail.

Law enforcement offices investigating computer related crime are often introduced to the case well after a criminal act has been discovered. The offices carrying out computer forensic investigators most frequently commence their activities part way along the crime to court path. The path shown in Figure 1 commences when the criminal act is committed and continues through to prosecuting the case in court. Figure 1 also shows the points along the path where the involvement of systems administration and law enforcement personnel start, overlap and fade into the background.

After a computer related criminal act is committed, it may be some time before anyone notices. The act may be first noticed by friends or family at home but as many acts are committed on networked computer systems in the work place, educational institutes, or public facilities such as Internet cafes, it is the systems administration staff that will probably have their curiosity attracted when they notice unusual activity. To satisfy this curiosity a period of observation would clarify whether further action is justified. Assuming that an act justifies further action it is reasonable to expect evidence to be sought to confirm something untoward is in progress. This leads to the suspicion being confirmed and at this point one could state that the criminal act has been discovered. It is at this point that someone will decide to either ignore and forget the matter, or to continue along the path and to collect evidence to support a response to the situation.

This is another critical point along the path because now the decision required is to either deal with the matter privately or to advise law enforcement authorities. Until this stage the responsibility for action resides with computer systems administration personnel and no law enforcement office is involved. Once a law enforcement agency is advised of the situation and they commence a crime investigation the situation changes dramatically. It is now imperative that activities comply with accepted practices and nothing is done to jeopardise a successful prosecution. Law enforcement offices may now seize digital data and identify and preserve evidence. From about this point along the path the systems administration staff fade into the background and law enforcement offices take control and responsibility for the case.

They will copy, analyse, and interpret the data before presenting the prosecution report in court.

Figure 1 shows that the systems administrator may be critical to the success of a prosecution case because their early recognition of a situation and their subsequent actions may either greatly assist or hinder the work done by law enforcement offices. Although they have distinctly separate and vastly different roles and responsibilities, by working in concert they may dramatically improve the chances of bringing about a successful result to a prosecution case.

Figure 1: Crime to Court Path

The danger of criminal activity not being successfully prosecuted due to the failing of a computer forensic process is very real. This is further exasperated by the provision of advice that works against the objectives of computer forensic investigators. Advice given by Bologna and Lindquist (1995) discusses how to use a computer, modem, communication software, procomm, tymnet and databases such as TRW and D&B. They then state that, "You are now ready to dig into files. The procedure is to (1) turn on the PC; (2) insert the communications software diskette in the A: drive; (3) when the program is loaded, dial up the database provider; (4) when connection is made, sign on with your user ID and then your code name; (5) when the menu is displayed, select an area of interest and follow instructions." (Bologna and Lindquist, 1995). Further, in discussions on forensic accounting of large computerised account systems Bologna fails to acknowledge computer forensic science in any way (Bologna and Lindquist, 1995). Anyone following this advice will seriously compromise any prospective computer forensic investigation because the very search for evidence will intrude and alter critical files. A primary principle of computer forensic investigation is to conduct any analysis of digital evidence on a replication of the original data after it has been gathered in such a manner that the original data is not contaminated or altered.

While an investigation may be considered successful because a conviction was gained many convictions result from the suspected criminal confessing to an accusation and the need to prepare and present conclusive evidence is not demanded. In some cases the computer evidence is collaborating or circumstantial evidence that supports, but is not essential to, a case. In a recent Perth murder case, the female suspect admitted killing her husband after evidence showed that she had visited two Web sites. One site was on how to hire a hit man and the other on how to dispose of a body. In other situations the perpetrator of the crime may be a professional and much respected computing expert familiar with how evidence may be obtained and too clever to be easily incriminated. In this type of case a more in-depth investigation may be required to produce evidence for prosecution and the evidence presented may include very technical concepts relating to sophisticated tools and systems that requires to be explained in non-technical terms so that members of the court will comprehend the issues.

The credibility of an expert witness may be crucial to the outcome of a case and where the volume of work and the increasing number of investigations demand quick results there may be a tendency to select new automated examination and analysis tools. Government agencies within the USA have approximately 1400 active cases of cyber crime being investigated and this number does not include the myriad of cases where computers have been seized and evidence gathered and analysed to support other crime cases (Hatcher, 2001). Under these circumstances there is a potential to discredit expert witnesses because as "point & click" wizards they may be perceived to have little or no expertise and not understand what they have done, nor why (Barbin and Patzakis, 2002).

The Role of Academia

Academia has an important role in meeting the needs raised by this increase in computer related crime and the subsequent investigations.

The aim of teaching computer forensics is primarily to meet industry demands by addressing the needs of law enforcement personnel and those that manage and operate computer systems. Academic research, teaching and training to support industry and law enforcement should improve confidence and credibility of investigators that result in better success rates of litigation which in turn may lead to crime reduction.

By comparing law enforcement requirements with existing solutions, the gaps in existing technology can be determined. By working together, researchers in academia, industry, and government can give our public servants the tools they need to address one of the critical public security and national security issues of the 21st century." (Vatis, 2002). This statement promotes the need by both law enforcement personnel and corporate system administrators to undertake training and education programs. Training in individual tools to improve competency skills and further education to enhance a broad understanding of concepts and processes related to subject that leads to gaining an academically recognised qualification. Together these help to improve the perception of being a creditable expert.

Content of a Computer Forensics Course

In order to meet this demand from industry a computer forensic course has been designed to suit both law enforcement offices and IT professionals. The objectives of the course include developing an understanding of the principles and practices of computer forensics. By the

end of the course students are expected to be able to understand and to practically demonstrate how to correctly access digital data equipment, obtain an exact working image, analyse the image and recognise evidence, then report their finding in an adversarial environment. The students are expected to maintain and document investigation notes, follow chain of evidence practices and have a thorough overview of computer forensic tools and the appropriateness of their use.

In its strictest and simplest sense one may argue that computer forensics is a highly technical task that belongs in the realm of the computer scientist. There are however many other aspects that need to be accommodated in order to achieve successful investigation and prosecution outcomes. Certainly an extensive knowledge of computer science is the foundation upon which the art is practiced. The ideal computer forensic expert witness is not only expected to conduct the scientific aspects of an investigation but also be able to present evidence successfully before a court of law.

This computer forensics course is a core unit in the Masters of Internet Security Management. It is undertaken in the final semester of full-time study. This is to ensure that students have a knowledge base of Masters level units that includes network and communication security, operating systems and software security, business intelligence and cyberwarfare, encryption, and Internet security. It is also included in other related Masters by coursework programs.

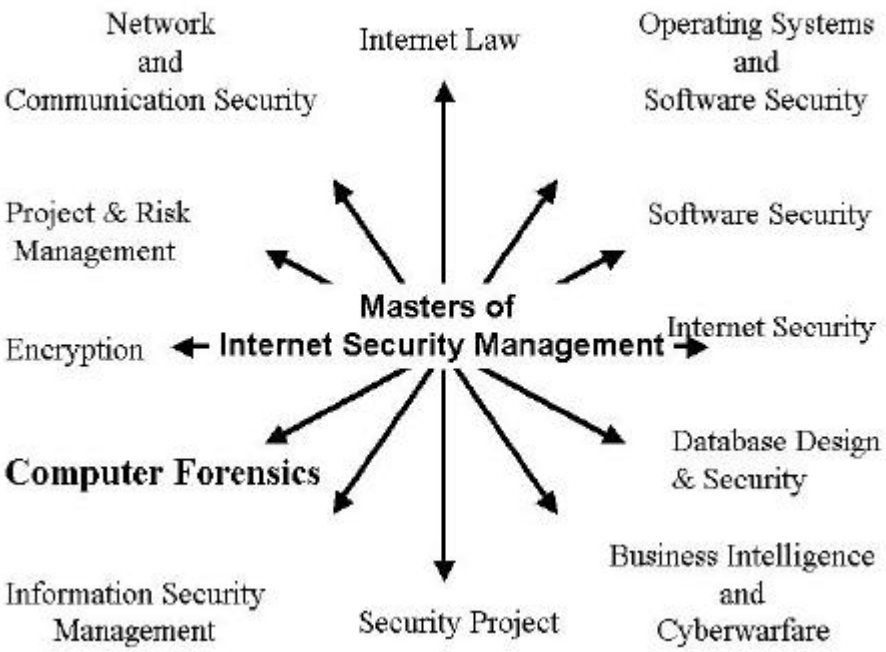

Figure 2: Core aspects of the Masters of Internet Security Management

The course addresses the points along the crime to court path. Students from law enforcement agencies backgrounds will be able to extend their skills by engaging in the concepts on which the practices they apply in the field are based. Students employed in systems administration areas will gain a perspective of law enforcement issues and the practical application of policing investigation practices. Students from these backgrounds would normally be expected to share their work place and field experiences for the benefit of the other students.

The content of the computer forensic course offered at Curtin University assumes a prior knowledge and understanding of network and communications security. Areas to be covered in the course include;

Introduction to Forensic Computing
Op Sys & file structures
Computer forensic technology tools and concepts
Computer, network and Internet architecture
Isolation and seizing of equipment and files
Computer image verification and authentication
Data recovery, imaging, and evidence options
Investigation techniques and processes
Discovery of evidence and chain of evidence
Law & Role of Expert Witness
National and international legal issues, rights and responsibilities

The course will be conducted over 12 weeks with each week consisting of three hours split between lectures, seminars and laboratory exercises. Students engage in practical exercises where computer forensic tools are used to capture, copy and analyses digital data so that the students gain an appreciation of the physical requirements of undertaking a computer forensic investigation. This is seen as a vital ingredient to the course. Currently, Curtin University academic staff are working closely with a number of law enforcement agencies on joint research projects and teaching programs. Students will investigate criminal cases taken from the public domain supported by evidence from enforcement agencies. Students are expected to take digital media and progress along the crime to court path to the extent where they will be expected to present their findings in a simulated adversarial environment.

The course will run for the first time in the second semester of 2003.

Conclusion

Law enforcement agencies are responsible for investigating and prosecuting breaches of the law. Computer related crime is increasing and the workload of investigators is growing.

New automated examination and analysis tools assist investigations but there is an established need to provide additional training and education programs plus a need for University research to support computer forensic education.

The crime to court path shows points along the way from the time that an act of crime is committed through to when it is dealt with by the courts. It shows where IT professionals may assist law enforcement offices and at which point crime investigators take up a case.

An explanation of the Curtin University Master of Internet Security program put into perspective how the computer forensic course relates to and addresses the needs of industry and law enforcement officers.

The computer forensic course offered at Curtin University is designed to meet industry needs by adopting a practical application of knowledge approach.

References

Armstrong, I. (2002) *Now in Session. The Judiciary and the Digital World,* SC Info Security Magazine. p19.

Barbin, D., Patzakis, J. (2002) *Computer Forensics Emerges as an Integral Component of an Enterprise Information Assurance Program,* Information Systems Control Journal, Volume 3, p25.

Bologna, G. J., Lindquist, R. J. (1995) *Fraud Auditing and Forensic Accounting (Second Edition),* John Wiley & Sons, Inc., New York.

Furnell, S. (2002).. *Cybercrime : Vandalizing the Information Society,* Addison-Wesley, London

Hatcher, T. (2001). *Survey: Costs of Computer Security Breaches Soar* . CNN.com, . Available on-line at : http://www.cnn.com/2001/TECH/internet/-3/12/csi.fbi.hacking.report/index.html . (12 March 2001)

Icove, D., Seger, K., VonStorch, W. (1995) *Computer Crime. A Crimefighter's Handbook,* O'Reilly & Associates, Inc, Sebastopol CA.

Kruse, W. G., Heiser, J. G. (2001) *Computer Forensics. Incident Response Essentials,* Addison-Wesley, Boston.

Powers, R. (2002) Computer Security Issues & Trends : *2002 CSI/FBI Computer Crime and Security Survey*, Computer Security Institute. Available on-line at : http://www.gocsi.com . (9 November 2002)

Vatis. (2000) . Statement of the Director, National Infrastructure Protection Center, Federal Bureau of Investigation on Cybercrime before the Senate Judiciary Committee Criminal Justice Oversight subcommittee and House Judiciary Committee, Crime Subcommittee, Washington, D.C. February 29, 2000 . Available on-line at : http://www.usdoj.gov/criminal/cybercrime/vatis.htm . (14 December 2001)

Vatis, Michael. A. (June 2002). *Law Enforcement Tools and Technologies for Investigating Cyber Attacks. A National Needs Assessment*. Institute for Security Technology Studies, at Dartmouth College, Hanover, NH.

A Tutoring System for IT Security Education

M. Schmitt, J. Hu, and Ch. Meinel

Department for Theoretical Computer Science and New Applications,
Research group "Institute for Telematics"
University of Trier, Germany,
E-mail: {schmitt,hu,meinel}@ti.uni-trier.de

Abstract

The Institute for Telematics of the University of Trier is currently developing a tutoring system that teaches knowledge of IT security. In contrast to other tutoring systems, exercises are not made in a restricted simulation environment but on a real Linux system. This approach allows the learner to apply his or her skills easily in practice. Depending on the type of user (e.g., end user or administrator), different lectures are provided. Interaction takes place via a web browser-based interface. It provides the user with various navigation facilities, help assistance, and statistics on the current status of processing.

Introduction

Modern IT systems are inherently vulnerable to hacker attacks. In the past, the '*I LOVE YOU*' virus and the '*CodeRed*' worm have demonstrated this impressively and the number of known security holes is increasing daily. Pitifully, many of these attacks succeed because people do not know about the vulnerabilities of their systems and how to defend against attacks. On the other hand, many risks could be avoided if both users and IT professionals were sensitized to and trained for IT security. Therefore, IT security education has become an important topic.
Recently, many universities have integrated lectures on computer security into their curricula. But besides knowing the theory, practical experience and the acquisition of practical skills are essential. If some tutorials are available, then these exercises are typically guided by an instructor.

There are also a few computer-based training systems that are concerned with IT security. The *ID-Tutor* (Rowe and Schiavo, 1998) and the intelligent tutoring system (ITS) described by Woo et al. (2002) familiarize the user with intrusion detection. The ID-Tutor creates audit files with information on user logins and executed commands. The user has to decide whether an intrusion has occurred, and, in case of an intrusion, he/she must resolve the problem. The ITS by Woo et al. is similar to the *ID-Tutor*, but it generates its missions from a knowledge base. With both tools, the users perform their exercises in a simulation environment. Unfortunately, for practical reasons, such a simulator can model a real system only to a very limited degree.

The Institute for Telematics, Trier, is developing a tutoring system for IT security, called *Lernplattform IT-Sicherheit* (LPF; English: learning platform IT security). The LPF provides knowledge about security technologies and tools tailored to specific user groups. It is based on web technologies so that users can interact with the tutoring system via a standard web browser. In addition, the LPF provides both a German and an English frontend.

ISSN 1445-3312 print/ISSN 1445-3347 online

In contrast to other tutoring systems, exercises are not performed in a restricted simulation environment but on a real system with standard tools. The tutoring system makes sure that the Linux environment is set up properly before it asks the user to perform an exercise, and checks the modifications by the user afterwards. This approach allows the user to apply his or her skills easily in practice.

The LPF is based on Linux. This operating system has been chosen because it is freely available and is increasingly gaining popularity. It also provides a vast number of open source security and hacker tools. This makes it possible to ship the LPF as a complete system that does not require additional commercial software. The LPF is based on SuSE Linux, the Apache web server, and the PHP and Perl script languages. For practical exercises, various security tools and auxiliary programs are used such as *OpenSSL*, the *Nmap* port scanner, the *Nessus* security scanner, the password cracker *John-the-Ripper*, and the *Snort* intrusion detection system.

The tutoring system does not only provide Linux-specific contents but also covers general security aspects. The range of topics includes cryptography and secure email, authentication, firewalls, intrusion detection, viruses, and security scanning. In order to follow the course, a user is required to have some rudimentary knowledge about Unix concepts (e.g., how to enter commands in a shell, how to create a subdirectory or delete a file).

The System Architecture

An *Intelligent Tutoring System* (*ITS*) is able to communicate with a user to assess his or her results and to teach users in a manner that is appropriate according to their knowledge and skills (Persché, 1997). The primary tasks of an ITS are the modeling of (1) the knowledge of the domain (domain model), (2) the user (student model), and (3) the pedagogical strategies (tutor model). Furthermore, the development of an ITS must focus on the creation of a powerful user interface (Sleeman and Brown, 1982). The LPF is designed according to these ideas. Its system architecture is shown in Figure 1.

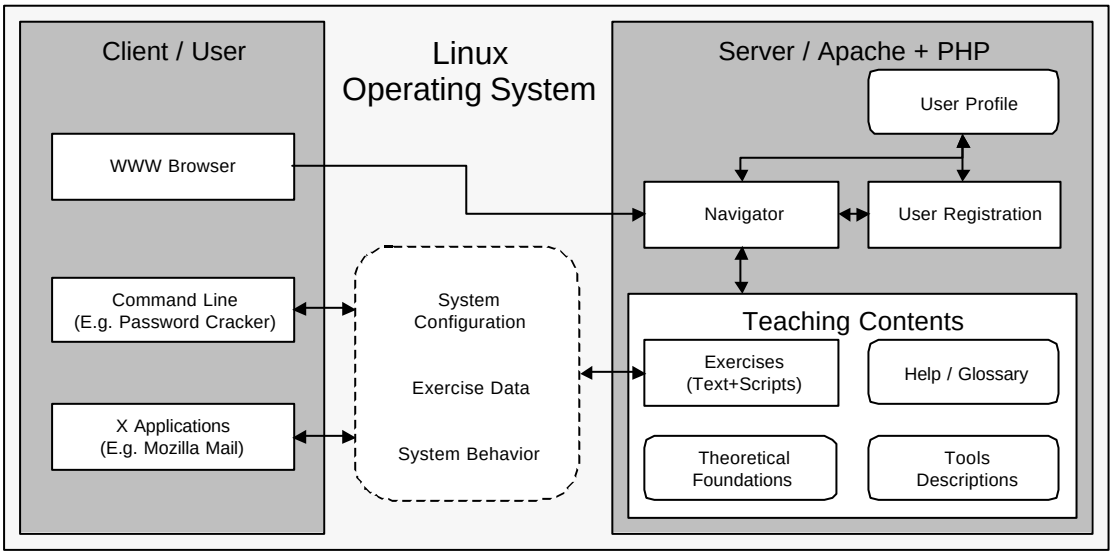

Figure 1: System Architecture

A web browser provides the student with a user-friendly and uniform interface. In order to perform the exercises, the user must execute some programs from within a command shell or run X window applications such as Mozilla Mail.

On the web server side (Apache with PHP module), the central logical component is the *navigator*. It is responsible for the interaction with the user and corresponds to the tutor model of the ITS concept. The *teaching contents* correspond to the domain model. They are represented by a collection of web pages and scripts. The *user profile* is the counterpart of the user model. It keeps track of the user's knowledge at every stage in the learning process.

Beside the web interface, the user and the tutoring system communicate indirectly via the Linux base system. This communication takes place implicitly by modifying the system configuration, creating exercise files and results, and observing the current system behavior.
In the following, the components of the tutoring system are discussed in detail.

The Navigator

The purpose of the navigator is to guide the user through the chapters. It also prepares exercises, assigns the exercises to the user, and processes user feedback. As the gateway of the core tutoring system, the navigator is able to track all communication with the user and analyze his or her learning process.

To present the teaching contents in a structured manner, the navigator creates linked web pages that are displayed in the web browser of the user (see Figure 2). The navigator also decides whether the user has finished a section successfully and where to continue at the end of a section.

When starting the LPF, the user must register or login with a valid account. Then, the navigator provides a list of available chapters from which the user can choose. When the user enters a chapter, the navigator creates a navigation bar on the left side of the web page that lists all sections and represents their type by a small icon (see Figure 2).

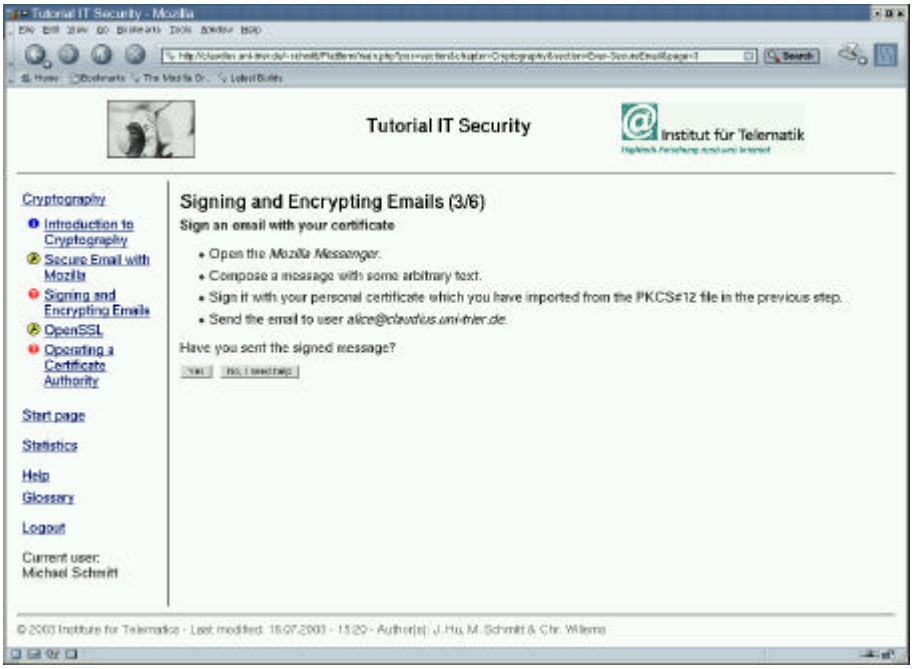

Figure 2. Exercise Example: Secure Email

The Teaching Contents

The teaching contents of the tutoring system fall into one of the following three categories:

- Theoretical knowledge on security concepts
- Technical descriptions of software tools and of the configuration of the operating system and applications
- Practical exercises based on the contents provided before
-

The first two types represent declarative knowledge that is presented to the user in hypertext form mixed with multimedia objects, such as graphics, images, or (Flash) animations. The exercises reflect procedural knowledge related to how a task is performed. The implementation of the procedural knowledge is much more complicated because it must be presented in terms of scenarios. This means that the steps and actions in the exercises must be planned carefully.

The teaching contents are organized into *sections*. A section is concerned with either theoretical concepts, tool usage, or one or more exercises. A section is also the basic unit for measuring the user's performance. Every section consists of one or more (web) *pages*. The exercise sections comprise some additional scripts.

Multiple sections are combined in a chapter that represents a security topic. In most cases, a chapter introduces some security concepts first. Then it explains some tools or commands. Afterwards, the user is asked to perform some practical exercises.

An important prerequisite for self-study is specific assistance. In case of problems, the user is supported by queries and tips for fault analysis. This help information is provided in a pop-up window as a series of questions and answers (see Figure 3). In addition, a glossary allows to lookup unknown acronyms or terms that are mentioned in the text.

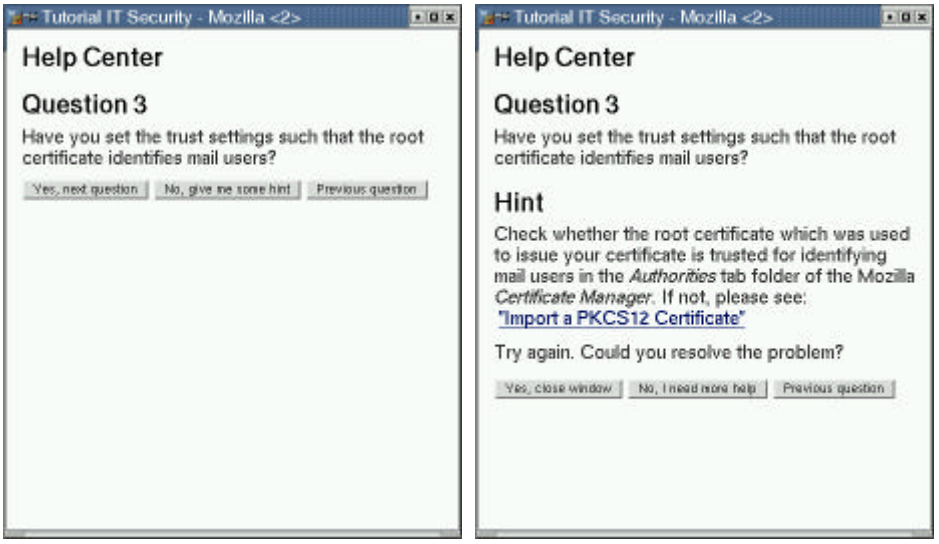

Figure 3: Help Information

Technically, for each section, there is a description file that provides the navigator with meta information, including the section type, the number of pages, the title, etc. A chapter description file specifies how a chapter is organized. Finally, a profile description file defines which chapters are reasonable for which type of user. Based on these three kinds of

description files, the navigator is able to construct a hierarchy of chapters and sections and to produce appropriate hyperlinks.

The use of description files makes it very easy to adapt user profiles and the structure of chapters in the future, without having to touch the individual sections. Instead, only items in the description files have to be changed for that purpose. Furthermore, it allows one to define several chapters that share some common sections.

Exercise Management

Exercises are specified as Perl and PHP scripts and interpreted by the navigator. An exercise takes place in three steps:

First, the working environment, i.e. the Linux operating system, is configured. For example, if the user is to perform security scans, a set of services are activated so that the user gets effective results.

The next phase deals with generating questions or tasks and passing them to the user. Where possible, these tasks are created dynamically, i.e., all users do the same type of exercise but with different detailed content. For example, for password cracking, a Unix *passwd* file (that the users must decrypt) is generated at run-time. Exercises must be created in such a way that the degree of difficulty does not vary. For instance, passwords created in the *passwd* file are selected randomly from the same dictionary and can be cracked in similar time.

After the user completes the task, the LPF evaluates the results. In some cases, it may also be necessary or useful to generate some background load *during* an exercise. For instance, the tutoring system might instantiate a *telnet* session while the user practices network sniffing so that some critical passwords can be observed in the data packets.

Example

The preparation, execution, and result analysis of exercises in a real system environment is a complicated task for which many technical details must be considered. The effort needed to set up a proper environment is illustrated by an exercise for exchanging confidential emails. Its purpose is to make users familiar with certificates so that they can sign and encrypt emails. The LPF user interface for this exercise is shown in Figure 2.

In order to prepare the working environment, the tutoring system clears all relevant settings made throughout former exercises or by other users and creates the necessary working directories. Then, a local mail server is set up (unless it has been set up before) and a virtual user *Alice* is created as a communication partner for the user. Her Linux account and email settings are configured. Secure communication based on asymmetric keys requires an authority that certifies that a public key indeed belongs to a particular user. The functionality of a simple certification authority (CA) is realized by a sequence of OpenSSL commands that are executed by the web server. Their execution is invisible to the user. Afterwards, the tutoring system issues certificates for Alice and the user and installs Alice's certificate. Finally, a web page is sent to the user that asks him/her to import the certificate into the mail client.

The generation of the exercise starts when the user confirms the import of his/her certificate and requests a new web page with further instructions. First, the tutoring system creates a random message. It is signed on behalf of Alice and sent to the user using the local mail

server. Then, the user is asked to (a) check the reception of the email, (b) verify the signature attached to the message, (c) accept and trust Alice's certificate, (d) reply to Alice's message with his or her own signature, and (e) send a message encrypted with Alice's certificate to Alice.

After the user has stated that he or she has successfully finished the exercise, his/her actions are evaluated. First, it is checked whether Alice has received a message and if this is true, the message is fetched from the mailbox. Next, its signature is verified to see whether it has been sent by the user and matches with the original message. Then, the message is decrypted with Alice's private key. If all checks are positive, the tutoring system records the successful completion of the exercise in the user profile. Depending on the outcome of the exercise (and if need be, the error case), a specific web page is sent to the user.

User Modeling

An important feature that distinguishes an ITS from other tutoring systems is that it “knows” each individual user and tracks his/her performance. In the LPF, this feature is supported by the *user profile*.

The user profile comprises two kinds of data: One is static personal information. It comprises name, password, and language. In addition, the user is assigned to one of (currently) three categories: administrators, end users, and students. Depending on the type of user, the requirements with regard to IT security may vary strongly. For instance, a typical end user needs information on the secure exchange of emails; but in contrast to a system administrator, he is not concerned with aspects of intrusion detection. Similarly, the depth in which a topic should be discussed may have to vary. The LPF takes this into account by compiling information and exercises individually for each user group.

The second class of data includes information on completed sections, the link to the last visited web page, help requests and their frequencies, and so on. In order to complete a topic, the user must pass all exercises in a chapter. A user can repeat an exercise until he/she finally finds a correct solution. The dynamic profile data are used to analyze the user's performance. The progress can be watched on a statistics page at any time. Besides an overview of which chapters have been visited/completed, the tutoring system tracks the time spent in each section. Among others, this information is intended to be used for optimizing the size and structure of the learning contents.

Technically, user profiles are implemented by a PHP class. When a user logs in or out, a user object is read from/written to a floppy disk. In addition, automatic storages take place whenever the user finishes a section or a period of time has elapsed. Thus, even if users corrupt the Linux systems by some exercise, their test results are still accessible.

Summary and Outlook

In this paper, we have presented a web-based tutoring system for IT security. The LPF improves the existing security education activities in several ways:

- It offers a real system environment instead of a limited simulation environment to the user.
- It provides a web browser-based user interface with hyperlinked teaching contents.
- It creates exercises dynamically.

The tools and programs needed in the exercises are available via the browser or are already installed on the system.

While the main framework of the LPF is completed, designing and implementing teaching material is an on-going process. Currently, five chapters are available. These chapters deal with authentication, secure email, certificates and PKIs, data security, network services, and port scanning.

Sometimes, the user needs to execute privileged commands to perform certain exercises. This introduces the risk that the user spoils the system. Since the LPF runs on the same operating system on which the user performs his/her exercises, this may lead to a complete break-down in the worst case scenario. Under such circumstances, the LPF must be reinstalled on hard disk. Currently, we work on two solutions to overcome this problem. The first is to ship the tutoring system as a Linux system that can be booted from CD and does not require any hard disk installation. As a second approach we investigate the use of virtual machines on top of a base Linux system (Dike, 2000). In this context, we also examine the possibility to provide the LPF as an online tutoring system that can be accessed remotely.

Furthermore, we are looking for a way to provide a pure browser-based user interface where external applications are displayed inside the browser. This would avoid the necessity to switch between different application windows during exercises. A possible solution is to embed a Java applet into the web page that connects to a Linux server (Cao et al., 2002).

References

Cao, J., Chan, A., Cao, W., and Yeung, C. (2002). Virtual Programming Lab for Online Distance Learning. In *Proceedings of the First International Conference*, ICWL 2002, pp 216-227, Hongkong, China.

Dike, J. (2000). A user-mode port of the Linux kernel. In *Proceedings of the 4th Annual Linux Showcase & Conference*, pp 63, Atlanta, GA. Usenix.

Persché, R. (1997). Immediate Feedback During Online Lectures. Master's thesis, Institute for Information Processing and Computer Supported New Media of the Graz University of Technology, Graz, Austria.

Rowe, N. C. and Schiavo, S. (1998). An Intelligent Tutor for Intrusion Detection on Computer System. *Computers and Education*, pp 395-404.

Sleeman, D. and Brown, J.S. (1982). *Intelligent Tutoring Systems.* Academic Press Ltd., London.

Woo, C., Choi, J., and Evens, M. (2002). Web-based ITS for Training System Managers on the Computer Intrusion. In *Proceedings of the 6th International conference* ITS 2002, pp 311-319, Biarritz, France and San Sebastian, Spain.

Using Outcomes-based Assessment as an Assurance Tool for Assurance Education

S. Older and S.K. Chin

Department of Electrical Engineering and Computer Science
Systems Assurance Institute
Syracuse University
Syracuse, NY 13244, USA
Email: sueo@ecs.syr.edu, skchin@syr.edu

Abstract

We discuss our efforts to deliver a graduate-level assurance curriculum with a strong emphasis on logic and formal methods. Specifically, we describe what we are teaching in two of our foundational courses, as well as what our students are learning. We also advocate the use of an outcomes-based approach when developing IA courses and curricula. We have found that focusing on the desired educational outcomes from the outset has made it easier to identify what is working and what is not, and we wish to share our experiences.

Introduction

The goal of Syracuse University's Certificate of Advanced Study in Systems Assurance (CASSA) program is to develop students who (1) comprehend the concepts underlying security and system assurance; (2) can apply those concepts to construct assured systems; and (3) can critically analyze and evaluate systems' conformance to their requirements. Because of this third requirement, a key component of the CASSA program is an emphasis on using formal mathematics and logic to provide a rigorous basis for the assurance of information and information systems. The CASSA program exists within the framework of the Computer Science and Computer Engineering Master's programs: students must satisfy all requirements of their home Master's program, as well as satisfying CASSA-specific requirements. In particular, all students must take a combination of courses that provides hands-on experience in both systems building and in using formal methods to analyze and evaluate system behavior.

Our purpose in writing this paper is twofold. The first is to report on our progress in delivering an Information Assurance (IA) curriculum with a strong emphasis on logic and formal methods. In (Older and Chin, 2002), we described our experiences in developing the CASSA program, as well as the challenges inherent in incorporating mathematical and logical rigor into an IA curriculum. Here, we provide more details about what we are doing in our courses, as well as our ongoing attempts to answer the following questions:

To what degree are we being successful? What are our students learning?

We discuss two specific courses that serve as elective courses in the CASSA program: *Modeling Concurrent Systems* (CIS 632) and *Principles of Network Security* (CSE 774). For each course, we describe the desired educational outcomes and what we have done to realize and measure those outcomes. We also discuss our observations of student achievement (both subjective and objective), and the changes we have made or intend to make as a result of

ISSN 1445-3312 print/ISSN 1445-3347 online

these observations. We hope that our experiences are informative for others interested in introducing formal methods and logical rigor into an IA curriculum.

The second and broader purpose is to advocate the use of an outcomes-based approach when developing IA courses and curricula. To be explicit, we do not wish to promote additional bureaucracy or the complex instrumentation of courses. Rather, we have found that focusing on desired educational outcomes from the start has made it easier to identify what is working and what is not, and we wish to share our experiences.

In our opinion, the outcomes-based approach is especially useful for developing IA curricula. IA is such a broad field that no individual program can cover everything. Focusing on desired outcomes helps identify which topics to include and in what depth. Furthermore, the greater specificity of desired outcomes allows for more detailed and precise discussions across disciplines. Ultimately, at the core of assurance is the mandate to guarantee with high confidence the quality of the resulting outcomes. We believe that we would be remiss if we failed to apply the underlying principles of assurance to all of our scholarly efforts. Just as we apply these principles with rigor to the design and evaluation of systems, we believe we should apply them to the design and assessment of our educational programs.

The rest of this article proceeds as follows. First, we describe the high-level educational outcomes of our CASSA program and how the requirements relate to these outcomes. We then illustrate how one of the high-level CASSA outcomes is addressed through two separate courses. The third section presents the course *Modeling Concurrent Systems*, an applied concurrency-theory course in which IA applications are an integral component. The fourth section focuses on *Principles of Network Security*: this course uses a variety of logical systems to describe, verify, and analyze properties related to network security. In the fifth section, we discuss our observations from these two courses, as well as some of the curricular adjustments that have arisen from these observations. We conclude with some final comments.

CASSA Educational Outcomes

The EECS Department has adopted an outcomes-based approach to curricula design (Syracuse 1998), in which we first formulate the outcomes we desire of our graduates and then use those outcomes to guide the development of rational curricula. This approach represents a shift from a traditional faculty-centered viewpoint of `*what do we teach?*' to a student-centered viewpoint of `*what do our students learn?*'

We made use of this approach in developing the CASSA program, focusing on our expectations for those students who successfully complete our program. No single curriculum can possibly address all of IA: concentrating on the desired educational outcomes helped us determine how to structure our program. In our case, the goal was to develop a coherent collection of courses to ensure the following outcomes:

1. Students comprehend the concepts underlying security and systems assurance.

2. Students can apply those concepts to construct assured systems.

3. Students can critically analyze and evaluate systems' conformance to their requirements.

Not surprisingly, these outcomes reflect the EECS department's long-standing emphasis on the use of mathematical and logical methods in engineering, computer science, and security.

These educational outcomes are broadly addressed by the CASSA program requirements. For example, to address Outcome 1, students must successfully complete both the Systems Assurance Seminar and a non-technical IA elective course (such as telecommunications policy, Internet law, or e-commerce). The seminar course serves as a gateway to more advanced assurance courses, introducing basic terminology and many of the nonmathematical issues related to information assurance. Furthermore, students must successfully complete a total of five courses from the *Foundations for Assurance* and *Assurance Applications* tracks, including at least two courses from each track. The *Foundations* track includes courses in cryptography, concurrency theory, and logical principles of network security. The *Applications* track includes courses such as Internet security, computer security, and security of wireless networks. Via this requirement, students are assured to receive hands-on experience both in building systems and in using formal methods, thus addressing Outcomes 2 and 3.

This setup raises an obvious question: what *specifically* can students do in the areas of constructing assured systems and critically analyzing and evaluating systems' conformance to their requirements? In this paper, we focus on the formal-methods aspect and describe our experiences assessing our students' learning in two separate *Foundations for Assurance* courses. We realize that, when the topics of outcomes and assessment are initially brought up, the typical reaction is resistance, due to a belief that it will be cumbersome, bureaucratic, and of little practical use. This has not been our experience. We have found that a good way to start is to follow Diamond's suggestion (1998, page 134):

> As an alternative to writing objectives in the abstract, ... [one can] develop strong, clear objectives by playing the role of the student and asking, "If I'm your student, what do I have to do to convince you that I'm where you want me to be at the end of this lesson, unit, or course?"

This approach is exemplified by the description of *Modeling Concurrent Systems* in the next section. One can also take a more structured approach suggested by Diamond (1998, page 132) and write objectives that include `a verb that describes an observable action' and `a description of the conditions under which the action takes place: "when given x, you will be able to ..." '. This approach is used in *Principles of Network Security*, which is described in the fourth section.

CIS 632: Modeling Concurrent Systems

The purpose of *Modeling Concurrent Systems* is to provide students with an in-depth understanding of the process-algebraic approach for specifying, modeling, and analyzing system behavior. Process algebras such as CSP (Hoare 1985) and CCS (Milner, 1980) provide a way to describe system behavior in terms of the events (i.e., abstract actions deemed *observable*) that can occur. The underlying theory also includes several useful notions of program equivalence and refinement, which are useful for compositional reasoning and analysis. There are automated and semi-automated tools available that allow one to apply the theory in practice to verify properties of nontrivial applications.

Educational Outcomes and Course Content for CIS 632

In the Fall of 2002, we focused on the use of CSP to specify and verify concurrent systems and to understand the emergent behavior of such systems. The educational outcomes appear in Table 1.

After completing this course, you should be able to:
• Use CSP or related calculi to write process descriptions and behavioral specifications.
• Use traces and failures refinement to analyze and relate system behaviors at multiple abstraction levels.
• Distinguish between the traces and failures models, and explain when each is appropriate to use.
• Use the model checker FDR to verify valid refinements or debug invalid refinements.
• Apply these techniques to specify and analyze nontrivial applications

Table 3: CIS 632 Educational Outcomes

We used Steve Schneider's CSP textbook (2000), supplemented with several exercises and examples from Bill Roscoe's textbook (1998). In the first portion of the semester, we introduce the basic operators of CSP, their operational semantics, and the trace and failures models, both of which induce notions of refinement. The trace model, for example, formally describes a process's behavior in terms of the set of traces—that is, sequences of events—that it can perform. These trace sets provide a basis for comparing, equating, and refining processes. Two processes are trace-equivalent when they have precisely the same sets of traces. A process Q refines P in the trace model (written $P \leq_T Q$) if every trace of Q is also a trace of P. Intuitively, if P corresponds to a specification of permissible behavior and Q refines P, then Q is guaranteed to exhibit only permissible behaviors.

Similarly, the failures model supports a notion of refinement based on a process's set of failures, which pair traces with sets of events. A process has the failure (t,X) if it can perform the trace t and then reach a state where its only possible actions involve events not in the set X. The failures model provides a finer notion of equivalence and refinement than the trace model does: it distinguishes processes not only by what they are able to do but also by what they are able to refuse to do. As a result, it supports reasoning about the potential for deadlock. A process Q refines P in the failures model (written $P \leq_{SF} Q$) if $P \leq_T Q$ and every failure of Q is also a failure of P.

We also introduced the model checker FDR2 (Formal Systems, 1997), which provides automated support for applying these notions in practice. The advantage of using a model checker is that one can analyze much larger systems than is possible by hand. In addition, model checkers can be useful in debugging designs: for example, when a desired refinement $P \leq_T Q$ fails, FDR2 produces a witness trace of Q that is not allowed by the specification P.

The final third of the course was spent on using CSP and FDR2 to analyze several different problems and protocols, such as the alternating-bit protocol and a distributed-database cache-coherency protocol. For three class periods, students worked in small groups of 4-5 students to specify and implement a one-lane bridge that safely supported bi-directional traffic. We also spent two lectures on the Needham-Schroeder key-exchange protocol and Gavin Lowe's (1996) use of CSP and FDR2 to uncover a previously unknown flaw in it. This analysis uses

CSP to model the system as the parallel composition of an initiator, responder, and intruder; CSP is also used to describe the desired authentication properties. FDR2 can then be used to try to validate refinement relationships between the system and the two desired properties. The result of FDR2's analysis is a witness trace that highlights a possible man-in-the-middle attack.

Assessment in CIS 632

Grades for this course were based on an equally weighted combination of homework assignments, quizzes, a final exam, and a final project. The purpose of homework was to keep students up-to-date with the material discussed in class and to familiarize them with the tools. The homework also served as simple formative-assessment tools: if several students had questions while working on the assignments or did poorly on particular questions, there was indication that certain topics needed to be reviewed again.

In total, there were six homework items. The first two homework items concentrated on sequential processes. The first homework required students to write CSP processes for a variety of scenarios, including one of their own choosing. For the second homework, students wrote machine-readable CSP and used the process animator ProBE (Formal Systems, 1998) to interact with their processes and to test their understanding of the CSP transition rules. The third and fourth homework items focused on the various parallel operators of CSP: the third homework assessed basic understanding of the operators, while the fourth homework challenged the students to use the operators to introduce constraints on a system. The final two homework items assessed students' understanding of the primary abstraction operators (i.e., hiding and renaming), as well as their mastery of the traces and failures models.

The quizzes and the final exam—all closed book and closed notes—served as summative - assessment tools, letting students demonstrate their understanding of the fundamental concepts. The quizzes were relatively lightweight and intended primarily as sanity checks. Typically, 60-75% of a quiz's points were for basic understanding of fundamentals (e.g., drawing transition graphs of processes, identifying a process's set of traces or failures, determining simple refinement relationships between processes). The remaining points tested students' deeper understanding of the concepts, such as writing a CSP process to model a scenario, validating or refuting claims about refinement relationships, and generating CSP processes that have (or fail to have) certain properties. The final exam placed more weight on these latter sort of questions: 34 points (out of 100) were for fundamentals, 30 were for writing CSP to model a scenario, and 36 were for validating/refuting a variety of claims about refinement relationships. Two examples of these claims follow:

- If $f(P) \leq_{SF} f(Q)$ and f is a renaming function, then $P \leq_{SF} Q$.
- For all processes P, Q, and R, and for all sets X and Y, the processes $P \underset{X}{\|} \left(Q \underset{Y}{\|} R \right)$ and $\left(P \underset{X}{\|} Q \right) \underset{Y}{\|} R$ are trace equivalent.

The first claim states that every failures-refinement relationship that holds between renamed processes (the process $f(P)$ behaves like P, except for a renaming of events via the function f) must also hold between the original processes. The second claim states that distinct parallel-composition operators are necessarily associative with one another. Both claims happen to be false.

On the final exam, students had very little difficulty with drawing transition graphs or identifying a process's traces and failures. The average grade (out of 10 students) for the combination of these questions was 85%, with a high mark of 100% (two students), a low of 62%, three students between 74% and 77%, and the rest between 85% and 97%. Students were also asked to write a CSP process to describe a banking account that allowed deposits, withdrawals (with limited overdraft protection), and queries, as well as requests to change the overdraft limit. Students then had to add a parallel constraint to the system to limit the number of rejected overdraft-change requests. On this question, the average grade was 74%, with a high of 93% and a low of 53%; five students received between 60% and 73%, and three received between 87% and 90%. The final question required students to judge the validity of various claims and to provide convincing explanations for their answers. Students had much more difficulty with this question: most students correctly identified the truth or falsity of the claims (worth a third of the points) but provided insufficiently convincing explanations. Here, the average was 63%, with a high of 89% (two students) and a low of 36%; the remaining students received grades between 39% and 86%, in a fairly even distribution.

The final project required students to use machine-readable CSP and FDR to model and reason about a nontrivial system or protocol. Students were required to submit annotated machine-readable CSP scripts containing descriptions of their system and of the desired behavioral properties, as well as the assertions (e.g., refinement or deadlock checks) necessary for validating those properties. Students were also required to submit 8-10 page project reports containing the following features: a high-level description of both the problem they were solving/analyzing and their CSP solution; a description of their use of refinements (e.g., explaining why trace refinements were used instead of failures refinements, or vice versa); and some analysis of their experience (e.g., unexpected results or design choices that were particularly good or bad). This paper was graded for grammar, spelling, and style, as well as for content. Students analyzed a variety of protocols and algorithms, including the Bully election algorithm, the Needham-Schroeder and TMN security protocols, the two-phase commit protocol for distributed transactions, and a distributed-sum algorithm.

CSE 774: Principles of Network Security

Principles of Network Security is an analytical course that uses predicate calculus, higher-order logic, and specialized logical systems to describe, specify, and verify the correctness and security properties of network security protocols, algorithms, and implementations. Students use formal logic to rigorously analyze cryptographic algorithms, key-distribution protocols, delegation, access control, electronic mail, and networks of certification authorities.

Because this course is fairly novel even among IA curricula and has no standard textbook, we describe its contents in more detail than the course in the previous section.

Educational Outcomes and Course Content for CSE 774

Both the educational outcomes and our discussion about this course make use of a modified Bloom's (1974) taxonomy to classify and distinguish the many kinds of knowledge and abilities. Some kinds of knowledge are at relatively low levels (e.g., *recalling* that $15_{16} = 21_{10}$), while others are at very high levels (e.g., *evaluating* whether an implementation meets a specification and requirement). The outcomes, which appear in the course's online syllabus, are listed in Table 2.

Comprehension

- Define the meaning of security services such as confidentiality, integrity, non-repudiation, and authentication
- Describe the characteristics of private key and secret key cryptographic systems
- Describe cryptographic algorithms for encryption, hashing, and signing
- Describe cryptographic protocols for session-based security such as Kerberos, and store-and-forward security such as Privacy Enhanced Mail
- Describe basic principles of trust topologies and networks of certification authorities

Application

- When given a block diagram or functional description of an implementation, you should be able to represent the implementation using predicate calculus
- When given protocol descriptions and trust hierarchies, you should be able to use specialized security calculi such as the logic of authentication for distributed systems to describe the protocol and trust relationships
- When given a trust topology, determine the necessary certificates for establishing trust in a key

Analysis

- When given a set of assumptions and a goal to prove, you should be able to prove, using formal inference rules, if the security goal is true or not
- When given a set of certificates, you should be able to formally derive whether a key is associated with a particular principal

Synthesis

- When given a description of a system or component and its specification and security properties, you should be able to construct a theory that describes both, and show if the security properties are supported
- When given a description of a trust topology, you should be able to create a formal description of certificates and trust relationships for the certification authorities

Evaluation

- When given a theory, inference rules, and a proof, you should be able to judge if the proof is correct
- When given a specification and implementation, you should be able to judge whether the implementation satisfies its specification

Table 2: CSE 774 Educational Outcomes

The first part of the course (outlined in Table 3) covers standard network-security fundamentals: basic security properties, cryptographic algorithms (e.g., DES and RSA), authentication, hash functions, digital signatures, protocols and certificates. The primary reference is William Stallings' classic text (1999).

Topic	Primary References
Basic Security Properties: confidentiality, authentication, integrity, non-repudiation, access control, availability; mechanisms; attacks	Stallings 1999 §1 Also: Saltzer et al. 1975, Lampson 1971
Conventional Encryption: DES, Electronic Code Book, Cipher Block Chaining	Stallings 1999 §3.1 - §3.3, §3.7
Confidentiality: placement of encryption, traffic confidentiality, key distribution	Stallings 1999 §5.1 - §5.3
Public-key Cryptography: principles of public-key cryptosystems, RSA, key management	Stallings 1999 §6.1 - §6.3
Message Authentication and Hash Functions: authentication requirements, authentication functions, message authentication codes, hash functions	Stallings 1999 §8.1 - §8.4
Hash Functions: Secure Hash Algorithm (SHA-1)	Stallings 1999 §9.2
Digital Signatures and Authentication Protocols: digital signatures, authentication protocols, digital signature standard	Stallings 1999 §10.1 - §10.3
Authentication Applications: Kerberos, X.509 authentication service	Stallings 1999 §11.1 - §11.2

Table 3: CSE 774 Topics in Security Fundamentals

The second part of the course (see Table 4) deals with reasoning about freshness of protocols, replay attacks, and role-based access control. We use the BAN logic (Burrows et al., 1990) to reason about freshness and potential replay attacks. We also introduce the formal definitions and properties of role-based access control (RBAC), as described by Ferraiolo and colleagues (1992, 1999, 2000). The knowledge expected of students is at the levels of comprehension, application, analysis and synthesis. Specifically, when given informal descriptions of key-exchange protocols, students are expected to be able to describe the protocol abstractly in the BAN logic; postulate initial beliefs about key associations, scope of authority, freshness of nonces, and so on; and show that, at the end of the protocol, belief in the distributed keys has been established. A similar set of skills is expected related to RBAC: when given an organizational structure of roles, students are expected to write down role-containment relations and derive the membership relations that are implied by a specific organizational structure.

Topic	Primary Reference
BAN Logic	Burrows et al. 1990
RBAC Definitions and Properties	Ferraiolo et al. 1999 (also: 1992, 2000)

Table 4: CSE 774 Topics in Belief Logics and Role-Based Access Control

The third part of the course (outlined in Table 5) focuses on authentication, delegation, and access control in distributed systems. Nine weeks is spent on this topic, the major focus of the course. The technical content of this part centers on the Abadi calculus (Abadi et al. 1993, Lampson et al. 1992) for reasoning about principals, their statements, and their beliefs.

Topic	Primary References
Underlying Semantics and Model	Howell and Kotz 2000 §1 - §3 Also: Abadi et al. 1993 §3.3 - §3.4
Axioms for Principals and Statements	Lampson et al. 1992 §3, Howell and Kotz 2000 §4.1 - §4.3 Also: Abadi et al. 1993 §3.1 - §3.2
Channels and Encryption	Lampson et al. 1992 §4
Group Names	Lampson et al. 1992 §5.3
Roles and Programs	Lampson et al. 1992 §6 Also: Howell and Kotz 2000 §4.4 - §4.5
Delegation	Abadi et al. 1993 §5 - §6.1 Also: Lampson et al. 1992 §7, Howell and Kotz 2000 §4.6
Interprocess Communication	Lampson et al. 1992 §8
Access-Control Decisions	Abadi et al. 1993 §6.2 Also: Lampson et al. 1992 §9
Reasoning about Credentials and Certificates	Wobber et al. 1994 §1 - §4.3
Extensions to the Logic	Howell and Kotz 2000 §6

Table 5: CSE 774 Topics in Authentication, Delegation, and Access Control

Assessment in CSE 774

Summative assessment is accomplished through four open-book, open-notes exams. Students receive written solutions as they turn in their exams. The questions are formulated with the educational outcomes at the various levels of knowledge previously described (comprehension, application, analysis, synthesis, and evaluation). While the course emphasizes the use of formal analysis, more weight is typically given to questions that ask students to set up the formal assumptions, definitions, and goals when given an informal problem description.

Fundamentals

The first exam occurred four weeks into the course. Two questions dealt specifically with comprehending cryptographic algorithms (DES and RSA): given particular inputs, students had to compute the outputs. Little class time was devoted to these topics, and the average grades on these questions (out of eighteen students) were 71% and 82%. At the levels of application, analysis, and synthesis, students were asked to model cryptographic algorithms in schemes such as electronic code book and cipher block chaining. Here, the results were less satisfying. For example, students were asked to model ECB encryption as a recursive function (where the particular encryption function and key are parameters) and prove that ECB inverts itself. In this case, the average score was 50%. However, the distribution of grades was bimodal: 25% of the students received full or close to full credit, 25% received no credit, and the remaining students received between 20% and 70%, with most of these students getting more than 50%. All of the students who received no credit had failed to take the prerequisite course in predicate calculus.

BAN Logic and Role-Based Access Control

The BAN material was also assessed in the first exam, at the application, analysis, and synthesis levels. Specifically, one question at the analytical level asked for a proof that, in the context of the X.509 protocol, one principal believed that another principal believed in a particular statement. The initial assumptions about keys and nonces were given. The average for this question was 76%, with seven students getting 100% and one student getting 0%.

Another question asked students to consider how beliefs would change if X.509 timestamps could not be checked accurately, and to justify their answers using the BAN logic. The average for this question was 70%: ten students got 100%, and three students got 0%.
RBAC was assessed in the second exam, which was given after seven weeks. By this time student enrolment had dropped from eighteen to fifteen. RBAC was assessed at the levels of application, analysis, and synthesis. Students were shown an organizational chart of company roles and asked to formally prove or disprove mutual exclusivity of various roles. The average on this problem was 66%, but the distribution was again bimodal. Five students earned 100%, six students earned between 70% and 95%, one student received 20%, and the remaining three students received 0%.

Authentication, Delegation, and Access Control

The underlying semantics of the principal calculus (Lampson et al. 1992, Abadi et al. 1993, Howell and Kotz 2000) is based on Kripke structures. A Kripke structure comprises a set W of *possible worlds*; an interpretation function I, which maps each propositional variable to a subset of W; and an interpretation function J, which maps each principal name to a binary relation over W. Intuitively, $I(p)$ is the set of worlds in which the propositional variable p is true, and $J(A)$ is the accessibility relation for principal A: if (w,w') is in $J(A)$, then principal A cannot distinguish between worlds w and w'.
Questions on the second and third exam focused on assessing the students' understanding at the comprehension and analysis levels. Specifically, they were given a particular Kripke structure and asked to evaluate the beliefs of principals in various worlds. The average grade on Kripke structures in the second exam was 58%, with three students receiving 100% and four students receiving 25% or less. These grades improved on the third exam to an average of 65% with five students receiving 100%, one student receiving 93%, and the remaining students receiving between 27% and 53%.

Students were also asked on both the second and third exams to prove an axiom of the calculus (e.g., `if $(A \wedge B)$ *says* s, then A *says* s and B *says* s'). The average grade for this question on the second exam was 48%, with two students receiving 100% and three students receiving 0%. On the third exam, grades improved to an average of 60%, with five students receiving 100%, one student receiving 80%, one student receiving 0%, and the rest receiving between 20% and 60%.

The fourth and final exam (given to fourteen students after fourteen weeks of class) focused on reasoning about certificates, delegations, and authority and on proving properties of credentials used in the Taos operating system (Wobber et al. 1994). This time, students were presented with a client/server system where the server receives a message `C_{Alice} *says RQ*' within the context of boot, delegation, and channel certificates. The students first had to set up a theory whereby the server could conclude the statement *((Machine as OS) for Alice) says RQ*, and then they had to carry out a formal proof to justify their conclusions. This problem was very similar to the extended example of (Lampson et al. 1992). The average score on these questions was 67%, with seven students scoring 80% and above (two at 100%, one at 97%, and four at 80%), and the remaining seven students with scores from 33% to 60%.

Observations and Adjustments

Based on our observations at the course level, we have identified both pedagogical principles and suggested adjustments, at both the course and curricular levels. We outline these findings here.

Course-level Observations for CIS 632

We have found it useful to introduce the automated tools early in the semester: it helps students view the process algebra as more like a programming language (and therefore *real*) than just a mathematical notation. While grading homework early in the semester, we noticed that students in the undergraduate version of the course seemed to write higher quality CSP code: the graduate students wrote code that was technically correct, but it had a very strong imperative flavor (e.g., large numbers of nested conditionals). We postulate that, because the undergraduates all had functional-programming experience, they were better prepared to think about computational solutions in a non-imperative way.

The results from the final exam suggest that students have good intuition about the concepts but need more practice in justifying answers rigorously: for example, many students would give a specific example to justify that a general claim was true.

We have also found that students have difficulty grasping the distinction between the *properties* one wishes to prove about a system and the CSP idiom of imposing *constraints* upon a system through parallel composition. The properties are described as CSP processes, and one can check whether the system satisfies those properties by checking whether a refinement relationship exists. Constraints are also represented by CSP processes, placed in parallel with the system to *enforce* certain properties. The confusion seems to arise because the same language is used both to express properties and to describe the system design: students have difficulty maintaining that distinction.

Having originally identified this problem in Fall 2001, we made a concerted effort in Fall 2002 to mediate it. In the in-class bridge project, two groups were charged with identifying the desirable properties (e.g., *all cars on the bridge are going in the same direction*) and describing them in CSP; the remaining groups were to model the bridge and a traffic-light system in accordance with a proposed solution. We would then, as a class, combine both parts in a single file and check the necessary refinement relationships. The expectation was that, by separating the tasks, students would concentrate on either properties or the system and be able to see how they relate to each other. However, both groups charged with writing properties repeatedly asked for clarification of their task: they had trouble comprehending how their properties were related to the overall system.

Consequently, we plan to use the process algebra CCS in the next offering of this course. The CCS notion of synchronization differs from the CSP notion in a way that avoids the constraints idiom. Furthermore, the available automated tools for CCS—such as the Concurrency Workbench (Cleaveland et al. 1993)— support the use of temporal logic for describing desirable properties. We believe that the use of different notations will help students distinguish between system-level descriptions and their behavioral properties.

Course-level Observations for CSE 774

In the Fall 2002 offering, we were interested in seeing how far and how deeply our students could learn to use the principal calculus of Abadi and colleagues. We made mid-course corrections after observing that many students were not facile with the use of discrete mathematics (sets, relations, algebraic properties). As a result, we devoted more time to reinforcing discrete mathematical notions at the expense of reasoning about additional protocols, such as electronic banking protocols. In the future, we plan on rectifying the lack of facility with discrete math by incorporating more applications such as RBAC and modal logic into the predicate-calculus course that is a prerequisite for this course.

In our view, limiting the time spent on cryptographic algorithms so as to devote time to the authentication logic was well worth it: students gained knowledge that enables them to think rigorously about authentication, certificates, delegation, and access control. Furthermore, the first-exam results demonstrate that students are capable of picking up the cryptographic material on their own.

Curricular-level Adjustments

The assessment results from both courses indicate that our students need practice in working simultaneously at two levels of abstraction: (1) they must be facile with the rules of a given logical system, and (2) they must be able to link those rules to a particular instance or application. In general, our students have demonstrated good intuition about concepts, but they lack experience in translating intuition into precise and rigorous statements that are amenable to analysis and verification.

In addition to the course-level changes, we are making curricular adjustments in our Computer Science (CS) and Computer Engineering (CE) programs to address these needs. First, we are redesigning our prerequisite courses to provide additional experience in relating the abstract to the concrete, so as to minimize the amount of remedial work necessary in advanced classes such as CIS 632 and CSE 774. Because the same person teaches both CSE 774 and its prerequisite course CSE 607 (*Logical Basis for Computing*), we were able to initiate this adjustment in Spring 2003.

CSE 607 is a required course for all CE Master's students. A primary objective of the course is to enable CE students to use predicate calculus to reason about specifications and implementations and how they are related, much as they use Laplace transforms to reason about communications systems. This course has a reputation among students as being hard (we suspect that a major reason is a lack of experience in relating the abstract and concrete), and a large percentage of students wait until their final semester to take it.

In Spring 2003, we introduced Kripke structures in CSE 607 as one application of the predicate calculus, set theory, and relations. Because this course is required for all CE students, the enrolment (79 students) was significantly higher and more diverse than that of CSE 774. Despite these apparent challenges, the average score on the Kripke-structure exam question was 68%, a marked increase from the average grade on the first Kripke-structure exam question in CSE 774. However, the average on this question was still lower than the average on the rest of the exam, which required students to state and prove properties about concrete relations (average grade of 75%), prove properties about an algebraic system (83%), and translate English arguments into predicate calculus and give a formal proof of validity (87%).

. The second curricular adjustment is larger in scale and in time horizon. We are moving towards a common core in both Computer Science and Computer Engineering that emphasizes logic, concurrency, and functional programming. In our view, knowledge of these three areas provides a platform for developing new theories and the means to specify and verify particular systems and applications. Logic provides a means for reasoning, concurrency is the building block for complex systems, and functional programming serves as a mechanized animation of these ways of thinking

Conclusions

We have described two courses that illustrate our efforts to introduce rigorous foundations for assurance into an IA curriculum. Specifically, we have presented not only the material that we cover, but also how we assess students' mastery of that material and how they are doing. The statistics that we have provided are not by themselves statistically significant. However, we believe they shed light both on how we are doing so far and on what is possible in a Master's-level IA curriculum. Furthermore, students have responded positively to these courses: in fact, multiple students have applied to our Master's CASSA program as a direct result of having analyzed security protocols in the concurrency course.

Based on our experiences, we advocate using an outcomes-based approach to develop courses and curricula. We are not assessment experts, but rather users who have found value in adopting practices put forward by experts in the field of Higher Education. We have tried to demonstrate that the process does not have to be painful or cumbersome. Even a lightweight approach as in CIS 632 offers a lot of benefits. A course's educational outcomes are suggestive of the appropriate assessment techniques, and both outcomes and assessment are in the instructor's control. Another advantage of writing outcomes is that information is communicated more precisely, both to students and to other faculty. Because we have thought about educational outcomes, several faculty across several courses have developed consensus regarding our Computing Engineering and Computer Science Master's programs. Because we have assessed our outcomes, our course and curricular changes are more precisely aimed at addressing shortcomings. We are moving towards a common core for these two programs that includes logic, functional programming, and concurrency. This common core would help prepare students for the two courses discussed here, as well as for the rest of our Master's programs.

Acknowledgements

This work is partially supported by the National Science Foundation under Grant Number DUE-0241856, as well as by the Information Assurance program of the New York State Center for Advanced Technology in Computer Applications and Software Engineering (CASE).

References

Abadi, M., Burrows, M, Lampson, B., Plotkin, G (1993) A Calculus For Access Control In Distributed Systems. *ACM Transactions on Programming Languages and Systems*, 15(4):706-734.

Burrows, M. Abadi, M., Needham, R (1990) *A Logic Of Authentication*. SRC Research Report 39, Systems Research Center, Digital Equipment Corporation, Palo Alto, CA.

Bloom, B.S. (1974) *The Taxonomy of Educational Objectives: Affective and Cognitive Domains*. David McKay, New York.

Cleaveland, R., Parrow, J., Steffen, B (1993) The Concurrency Workbench: A Semantic-Based Tool For The Verification Of Concurrent Systems. *ACM Transactions on Programming Languages and Systems*, **15**:36-72.

Diamond, R.M. (1998) *Designing & Assessing Courses & Curricula: A Practical Guide*. Jossey-Boss, revised edition.

Ferraiolo, D.F., Barkley, J.F., Kuhn, D.R. (1999) A Role-Based Access Control Model And Reference Implementation Within A Corporate Intranet. *ACM Transactions on Information and Systems Security*, **2**,1: 34-64.

Ferraiolo, D.F., Kuhn, D.R. (1992) Role Based Access Control. In *Proceedings of 15th Annual Conference on National Computer Security*, Gaithersburg, MD, pp. 554-563. National Institute of Standards and Technology.

Formal Systems (Europe) Ltd (1997) *Failures-Divergence Refinement: FDR2 User Manual*, Oxford.

Formal Systems (Europe) Ltd (1998) *Process Behaviour Explorer: ProBE User Manual*, Oxford.

Ferraiolo, D.F, Sandhu, R., Gavrila, S., Kuhn, D.R., Chandramouli, R. (2000) *A Proposed Standard For Role-Based Access Control*. Technical report, National Institute of Standards and Technology.

Howell, R., Kotz, D. (2000) *A Formal Semantics For Spki.* Technical Report TR 2000-363, Dept. of Computer Science, Dartmouth College, Hanover, NH.

Hoare, C.A.R. (1985) *Communicating Sequential Processes*. Series in Computer Science. Prentice Hall, London.

Lampson, B., Abadi, M., Burrows, M., Wobber, E. (1992) Authentication In Distributed Systems: Theory And Practice. *ACM Transactions on Computer Systems*, **10**,4: 265-310.

Lampson, B. (1971) Protection. In *Proceedings of the 5th Princeton Conference on Information Sciences and Systems*, Princeton, NJ.

Lowe, G. (1996) Breaking And Fixing The Needham-Schroeder Public-Key Protocol Using FDR. *Software Concepts and Tools*, **17**:93-102.

R. Milner (1980) *A Calculus of Communicating Systems*, volume 92 of Lecture Notes in Computer Science. Springer-Verlag.

Older, S., Chin, S. (2002) Building A Rigorous Foundation For Assurance Into Information Assurance Education. In *Proceedings of 6th National Colloquium for Information Systems Security Education*, volume 1. George Washington University Journal of Information Security.

Roscoe, A.W. (1998) *The Theory and Practice of Concurrency*. Series in Computer Science. Prentice Hall, London.

Schneider, S. (2000) *Concurrent and Real-Time Systems: The CSP Approach*. John Wiley & Sons.

Saltzer, J., Schroeder, M (1975) The Protection Of Information In Computer Systems. *Proceedings of the IEEE*, **63**,9:1278-1308.

Stallings, W (1999) *Cryptography and Network Security*, Second Edition. Prentice-Hall.

Syracuse University Department of Electrical Engineering and Computer Science (1998). *Developing Curricula To Meet The Needs Of The Next Millenium: Preliminary Report Of The EECS Curriculum & Course Development Committee*.

Wobber, E., Abadi, M., Burrows, M., Lampson, B. (1994) Authentication In The Taos Operating System. *ACM Transactions on Computer Systems*, **12**,1: 3-32.

If You Go Down to the Internet Today – Deceptive Honeypots

S. Yek[1] and C. Valli[2]

School of Computer and Information Science
Edith Cowan University
[1]E-mail: s.yek@student.ecu.edu.au
[2]E-mail: c.valli@ecu.edu.au

Abstract

This is preliminary research into the effectiveness of deceptive defensive measures in particular honeypots that use deceit as a primary defensive and offensive mechanism. Initial research has been conducted using the Deception Tool Kit and its ability to fool commonly available network scanning tools such as Nessus and Nmap The preliminary research indicates that these deceptive tools have a place in modern network defence architecture.

Introduction

Attacking trends over the last 5 years have shown Internet connections to be the increasingly cited point of attack (Power, 2002). This challenges the prior conception that most attacks are internal. While inside attacks still show significant numbers, the growth in reported out-sourced attacks show up to 60% on WWW/Company sites (*ibid*, 2002).

Statistics show 90% of respondents detected computer security breaches with financial losses within the last 12 months. Furthermore up to 40% detected Denial of Service (DoS) attacks (Power, 2002). These attacks indicate the mounting concern of defacing company reputation combined with theft of proprietary information, and financial fraud.

The goal of this project was to determine the ability for trapping and analysing the results of potentially dangerous attacks on a server when using a honeypot as the prime forensic gathering tool. For this purpose, a honeypot will then be defined as a 'resource whose value is in being attacked or compromised' (Spitzner, 2002).

These experiments were carried out in a private and secluded network of eight workstations and a server within the University. The victim machine was running Linux Redhat 7.2 operating system (OS) and had the DTK installed. The attack PC was primarily a Linux Redhat 7.2 machine, a Windows2000 PC was used to confirm results from the Linux tests with Nmap as a cross platform tool. The DTK itself used several deceptive operating systems with decoy port addresses and outputs to re-direct probes to the DTK. Information regarding the attacks were recorded via the logging features of the DTK and the conventional syslogd daemon. These results were then analysed to deduce the level of effectiveness of the honeypot in a real life situation.

Why use a deceptive honeypot?

The DTK has the ability to deceptively mimic the following operating systems: Windows NT, Linux, HP-UX, SCO Unix, SGI, IBM AIX, Sun Solaris, SunOS and

ISSN 1445-3312 print/ISSN 1445-3347 online

Ultrix. The deception toolkit has been used as the primary architecture of the honeypot. Firstly it was designed to be used as a defensive tool that systems administrators could use to defend systems. The DTK mimics industry standard servers and the services they provide by listening for inputs and redirecting traffic to customisable PERL script files. These script files then respond as an installed server or daemon should when sent commands that are legitimate or otherwise, selected

By deploying bogus services the machine will appear to contain seemingly numerous useful ports, and it will output responses that are intended to appear typical of a functioning server. While doing this, the DTK will record the actions of the intruder through log files that can then be analysed to determine the modus operandi of the attacker. The deceptive honeypot is intended to extend the time detection window as the attacker is drawn into probing services that are digital chameleons that have no real payload or substance.

The data collected from honeypot testing is 'normally of high value' (Spitzner, 2002). This is because information extracted from the analysis can be easily collected, organised, and documented. The high value information includes network activity and movements of the attacker, once in the system. Additionally, honeypots only capture information that is targeted to it. Therefore there is no overwhelming bandwidth or activity to overlook network progress by dropping packets, and potential attacks (Spitzner, 2002). Consequently there is a more efficient use of resources to provide manageable amounts of useful data for providing attack intelligence to the defender.

The honeypot was designed to act as a fully functional mail server installed within a typical online business running SSH, SMTP and POP3 services. For optimal operation in a real life situation, the honeypot would be required to be set up in an independent location away from the legitimate servers or within a tightly controlled and protected DMZ (Demilitarised Zone).

Design of the honeypot involves allocating inactive ports to a potentially viable host. This is intended to deceive the attacker into thinking they can receive valuable information from scanning port traffic and determining where that port connects to and identifying flaws in the movement of traffic (McClure *et al*, 2002).

The intended function of the honeypot is to confuse and disorientate the attacker by falsely directing them through bogus host lines that may or may not provide information that appears to be informative or even important. As there will seem to be numerous available ports, scanning will take considerably more time, depending on the configuration of the honeypot, and consequential disorientation may result in one of two ways. Firstly, the attacker will be bored, confused or angered and will cease hacking attempts. Secondly, the attacker will believe they have hacked into the system server, and will believe they have received valuable information.

In either case, the honeypot will have achieved its desired purpose in keeping record of the actions taken by the attacker unknowingly, and their consequent attacking strategies.

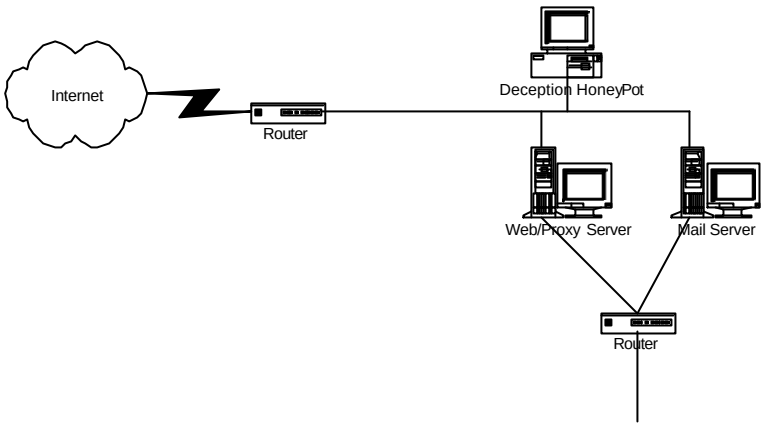

Diagram 1 - Honeypot deployment

Diagram 1 depicts a situation where a potential intruder enters through an Internet connection to the company router. The router has a connection to all network stations, including the honeypot. It then automatically directs the intruder to the honeypot, and away from genuine company assets. Similarly, an internal router directs traffic from within the company network without interference to the honeypot.

Methodology

There are various types of system testing and penetrating tools freely available to download from the World Wide Web. Two popular attacking softwares chosen were Nessus and Nmap. Both of these tools have won various industry accolades for software innovation and best of breed.

Nessus is a popular network security scanning and auditing tool (Insecure, 2002). Nessus checks for vulnerable systems by detecting all the ports running any given service and then probes and tests their security against known vulnerabilities. Nessus uses a server/daemon: nessusd, and a client nessus. Nessusd monitors the attacks and locates the security holes, then reports them to the Nessus client. The client then interfaces with the user and displays the results.

The results show information on which ports were scanned and corresponding responses. The client output references to possible security holes and exposure. More importantly, for the scope of this research, buffer overflows on ports suggest points of entry and manipulation to malicious users to initiate a DoS attack as a subsequent vulnerability.

Nmap (Network Mapper) is a freely available utility used for network exploration or security auditing. By using a technique known as OS fingerprinting (Fyodor, 1998) which examines returned IP packets received from the host Nmap is able to determine hosts available on the network, ports used, any packet filters or firewalls in use and what operating systems and versions are in use.

The attacks were performed through specifying ranges of ports to scan. These may be upon the assumption that the potential outside intruder has already performed some form of systematic fingerprinting, which is a tactic used to obtain company profiles of domain names, network blocks, and individual IP addresses connected to the Internet (McClure et al, 2001). Alternatively, the assumption also can be that the intruder may

be internal, where the IP addresses are already known, or easily accessible. Though not the highest, this is also a common source for attacks (Power, 2002).

Once IP address ranges are known, the intruder will perform port scanning in order to determine live hosts. This is often time consuming on the attackers part and is not entirely conclusive or accurate (McClure et al, 2002). The information that a potential intruder will receive can range from complete disclosure of the system's makeup including operating system (OS), network configuration and loaded services. This then allows the attacker to identifying related OS and service vulnerabilities. As many security vulnerabilities and exploits are dependent on the OS version an attacker can easily adjust their code to attack those weaknesses (Fyodor, 1998).

Through Nessus, brute force attacks and covert methods were used. The DTK was implemented on the victim host. Nessus was chosen as the software to initiate attacks against the victim host running the DTK utilising the brute force modes of Nessus. The thoroughness of the results of the attacks can then be compared to the actual logged information taken from the DTK log files and the standard syslogd facilities on the Linux system. Thus the honeypot DTK showed its level of effectiveness in distracting the attacker from real port addresses and its potential to prevent hazardous damage.

Limitations on the research are that the seclusion of the experimental network does not connect to the Internet and World Wide Web. This honeypot was designed to act as a fully functional mail server installed and would operate with SSH, SMTP and POP3 services.

Testing and Evaluation Plan

Nessus was used to brute force the DTK in each of its deceptive OS's. The probed ports were 1 - 1024, 12345, 1246, 2049, 5999 - 8000, 10000 – 28000. The Maximum number of threads was set to 8 and TCP connect scans were used to probe the ports on the victim host.

When the scan was complete a report was generated by Nessus with any detected security warnings and associated notes. A sample follows for the SMTP service when using SGI deception.

> **Warning found on port smtp (25/tcp)**
> The remote STMP server seems to allow remote users to send mail anonymously by providing a too long argument to the HELO command (more than 1024 chars). This problem may allow bad guys to send hate mail, or threatening mail using your server and keep their anonymity.
> Risk factor : Low.
> Solution : If you are using sendmail, upgrade to version 8.9.x. If you do not run sendmail, contact
> your vendor.
> CVE : CAN-1999-0098
> **Information found on port smtp (25/tcp)**
> Remote SMTP server banner :
> netsec.ecu SGI ESMTP Sendmail 8.1.2/8.1.3

Analysis of Results

When all scans were complete on each deceptive OS, the log files that the DTK generated for each OS were then imported into Excel spreadsheets for further viewing and analysis. Filtered data was retrieved and sorted into the spreadsheet. It was then evaluated by simply counting and recording buffer overflows on each of the probed OS's.

Ports	AIX	SGI	SUN	ULTRIX
19	5	2		2
25	94	31	2	32
110	75	24	5	26
365	5	2		2
893	5	2		2
2049	6	2		2
5999	6	2		2
6001	6	2		2
8000	6	2		2
10000	1			
12345	4	2		2
12346	4	2		2

Table 1 – Deceptive Buffer Overflows by Port/Service

Many security holes publicised are due to buffer overflows as a form of attack on company servers (Graham, 2000). Therefore it is a common problem faced and is a notable response from the Nessus reports. The number of buffer overflow indicates the number of times the DTK was able to output a red herring to intruders. Where an overflow of data is normally generated, there is a high likelihood that the program will crash or give an intruder root or high level access or privilege to a system.

Nessus believed it detected the following problems with the various deceptive operating systems

	Security			Rating of Problems			
	Holes	**Warn**	**Notes**	**Serious**	**High**	**Med**	**Low**
LINUX	1	5	3	0	50	50	0
NT	0	7	5	0	0	0	100
SOLARIS	0	6	4	0	0	0	100
HPUX	0	8	5	0	0	12	88
SUNOS	0	7	5	0	0	0	100
AIX	0	7	5	0	0	0	100

Table 2 - Nessus Results No Dangerous Plugins Used

	Security			Rating of Problems			
	Holes	**Warn**	**Notes**	**Serious**	**High**	**Med**	**Low**
LINUX	1	5	3	0	50	50	
NT	0	8	5	0	0	0	100
SOLARIS	0	6	4	0	0	0	100
HPUX	1	8	5	0	0	12	88
SUNOS	0	8	5	0	0	0	100
AIX	0	8	5	0	0	0	100
SGI	1	8	5	0	0	12	88
ULTRIX	1	8	5	0	0	12	8
SCO	0	7	5	0	0	0	100

Table 3 - Nessus Results Dangerous Plugins Used

A naïve hacker or script kiddie would typically rely heavily ontools such as Nessus to provide them with potential targets that they could compromise (Conry-Murray, 2001). This reliance on pre-compiled tools where they do not have to understand or be able to manually execute the attack is a hole in their offensive strategy. This enables the defenders to extend the detection window for the attacker as they are literally shadow boxing in a deceptive honeypot while leaving forensic trails of their activities.

Although the detected problems were rated as low they still leave an opportunity for a hacker to attempt to attack the system.

The SMTP service was the service that provided the most deceptive information and demonstrated the highest level of buffer overflows to a would be attacker. All of the deceptive OS implementation provided faked remote banners that took the form of netsec.ecu (Deceptive OS) SMTP Sendmail 8.1.2/8.1.3. In hacking guides (Anonymous, 2002; Fadia, 2002) commonly available on the Internet that target the SMTP service and in particular the Sendmail program, this information is used to determine what sort of attack the attacker should deploy to penetrate or dupe the system. This is some of the initial intelligence gathering that an attacker would undertake. Based on that knowledge they would then attempt various exploits on the system. Nessus believed it perpetrated buffer overflows that resulted in denial of service or allowed the successful anonymous relay of mail. It also further believed that the mail server was an open mail relay (Finlay et al, 2001; Rosenthal, 2002) which allows malicious attackers to pass mail through the server to other persons.

The SSH service in deception mode mimicked buffer overflows and core dumps whereby the attacker would once again believe they would have performed a successful denial of service on the SSH daemon. The POP3 service also gave away banner information which a hacker could use in the same manner as the SMTP banner to search for vulnerabilities.

All of the deceptive OS's demonstrated to Nessus vulnerabilities that simply did not exist in any real form on the victim system. This further confirms the veracity of claims (Cohen, 1998) 'The net effect is that attack tools that automatically scan for known vulnerabilities find what appear to be large volumes of vulnerabilities. When the attacker tries to interpret the results of automated scans, there is not enough

information to tell which of the detected vulnerabilities are real, and the number of detected vulnerabilities is very high and dominated by deceptions.'

One of the problems encountered was the DTK's inability to counteract Nmap OS fingerprinting techniques. The DTK returned consistently inconclusive results guessing that the OS was Standard: Solaris 2.x, Linux 2.1.???, Linux 2.2, MacOS. This gives the attacker a choice of four OS's to choose from or would potentially be used as fingerprint of a DTK by a wily attacker. This would pose a potential weakness when multi-homing sites on the one system using network aliasing.

Conclusion

The use of deception for defence has been around since the dawn of time and will be here from some time to come. Its place in a network defensive strategy is still relatively unclear. The use of deception based systems such as the DTK has the ability to fool many of the common scanners used by naïve or inexperienced attackers. This then leaves the naïve hacker at the mercy of their ignorance in successfully attacking the real system that in turn provides the defender time to instigate countermeasures to prevent further attack, or redirect further attempted incursions.

The DTK as tested provided extensive forensic data in its log files and to the syslogd functions on the attacked system. This forensic data would aid greatly in the investigation of an attempted break in.

The use of deceptive honeypots has weaknesses that need examination and further resolution if they are to be effective as a defensive mechanism. With the advent of multi-homing on one system/interface the ability to deceptively portray that interface as multiple systems/interfaces is an important extension of a deceptive honeypot to cope with modern networking technologies. Whether a more intricate deception that is more detailed and descriptive will aid in increasing the deception needs further investigation.

References

Cohen, F. (1998) A note on the role of deception in information protection. *Computers & Security*, **17**(6), pp.483-506.

Conry-Murray, A. (2001) Network security's not-so-secret ingredients. *Network Magazine*, **16**(8).

Fadia, A. (2002) Sendmail and Beyond: Kewl Tips and Tricks, (*Hacking Truths*), Available: http://hackingtruths.box.sk/smtp.htm [Accessed: June 2002].

Finlay, I., King, B., & Hernan, S. (2001) CERT® Incident Note IN-2001-02 - Open mail relays used to deliver "Hybris Worm": CERT.

Fyodor. (1998) Remote OS detection via TCP/IP stack fingerprinting, (*Insecure: Nmap*), Available: http://www.insecure.org/nmap/nmap-fingerprinting-article.txt (Accessed: May 2002).

Graham, R (2000) FAQ: Network Intrusion Detection Systems (*RobertGraham.com Infosec pubs*). Available: http://www.robertgraham.com/pubs/network-intrusion-detection.html#11 [Accessed: May 2002].

McClure, S., Scambray, J. & Kurtz. G. (2001) *Hacking Exposed: Network Security Secrets and Solutions* (3rd ed.): McGraw Hill, USA.

Power, R. (2002) Computer Security Issues and Trends. *CSI/FBI Computer Crime and Security Survey* **8**(1).

Raven. (2002) Sendmail, (*Hacking*), Available: http://www.astalavista.com/archive/hacking/sendmail.htm [Accessed: June 2002].

Rosenthal, C. (2001) What is Third-Party Mail Relay?, (*Maps: Transport Security Initiative*), Available: http://mail-abuse.org/tsi/ar-what.html [Accessed: June 2002].

Spitzner, L. (2002) Honeypots: Definitions and Value of Honeypots, (*Lance's Security Papers*), Available: http://www.enteract.com/~lspitz/honeypot.html [Accessed: April 2002].

Accessing Security Incident Information on the Internet

M.A. Belsis[1], A. N. Godwin[2], L. Smalov[3]

Data Knowledge Engineering Research Group
Coventry University, Priory Street, Coventry, U.K.
Tel. +44 24 76 631313, Fax: +44 24 76 888052,
E-mail: {Belsis, a.n.godwin, l.smalov} @coventry.ac.uk

Abstract

Computer security Incident Response teams have emerged due to the increase of computer crime. These can be national, international or organisation based. Maintaining a CSIRT poses a number of problems. In this paper the authors describe two of the technical problems that CSIRT's have, the storage and the acquisition of incident data. The paper describes a system based on the CORBA model that can be used for the efficient management of the incident recording database. The proposal also provides for alternative ways of accessing the database by companies and security analysts.

Introduction

The current rise of computer crime has sought the need for better information security (power 2000) and (Icove et al 1995). To assist organisations Computer Security Incident Response Teams (CSIRTs) have been assembled. The job of these teams is to receive information about computer security incidents that take place, analyse them and propose solutions to organisations. Some of the CSIRTs go further and assist companies in identifying the perpetrators and prosecuting them. CSIRTs use large databases that record details of security incidents coming from a wide variety and size organisations. They need to maintain information on this experience to develop defensive strategies, as described in (Anderson 1994).

Delivering a working CSIRT includes providing solutions to a number of problems. Some of these problems involve the management of the team. Examples include deciding on the boundary, the structure and the policy the team will follow. Currently numerous documents have been written on the way a CSIRT can be developed (NIST 1991), (West-Brown et al 1998) and (Brownlee 1998). These documents assist the teams in developing an overall policy and structure. Problems at a technical level include the acquisition, storage and analysis of security incident related information.

This paper discusses the problems that are inherent in the acquisition and storage of security incident related information. These problems are briefly explained and short descriptions of current solutions are provided. Although there is much discussion on the design of common security incident data models there is little discussion on the design of systems to support the proposed model. This paper proposes the use of the CORBA model in the process of acquiring security incidents related information.

Security Incident Reporting Structures

A number of CSIRT's exist; examples of such are the CERT/CC, CIAC and the CIRDB from CERIAS Laboratory. Each of them has developed and used their own data model to organize the reported security incidents. Generally these concentrated on storing the technical details

ISSN 1445-3312 print/ISSN 1445-3347 online

of an attack. Modern trends in hacking involve numerous hosts and/or networks located around the world (e.g. distributed denial of service attack). To be able to trace information about attackers involved in such attacks CSIRT's need to collaborate (Bouguettaya et al 1999a and 1999b). In order for the CSIRT's to collaborate they need to exchange precise incident information. The current incident recording structures in use this can be extremely difficult. CSIRT's store different types of information relating to security incidents. This means that two CSIRT's will often exchange information by telephone, fax or email.

Currently a lot of work is being done in developing common data models of security incidents and/or attacks. An example of such is the European proposal, Project *S2003*, which proposes a simple incident data structure and gives guidelines on its use in developing a comprehensive library of security incidents (Commission of the European Communities Security Investigations Projects 1992). The proposed model can be used by European Computer Security Incident Response Teams (CSIRT) as the means of storing data collected from security incidents.

An interesting approach is the Incident Object Description and Exchange Format (IODEF) developed by the Incident Taxonomy and Description Working Group (TF-CSIRT) (Demchenko 2001). The model was created to assist CSIRT's to exchange incident data. The model is based on the IDEFDM (Debar 2000) model and so it can automatically read incident information from any intrusion detection sensor that uses the IDEFDM model through the use of XML. Although the model was first created as a mechanism for information exchange it can be also used to develop a CSIRT Database.

The authors of this paper have delivered their own proposal in this area (Belsis et al 2002). This structure is part of work in progress aiming to provide a General Enterprise IT Security Data Model.

Deciding on a common agreed structure is not a complete solution. The common structure will solve only some of the problems that CSIRT's have. Other issues are concerned with the way this structure will be used, in particular the way data are acquired, protected and retrieved.

Issues on Secuirty Incidents Reporting and Retrieval

The security incident data model design should provide sufficient scope for the storage of data to satisfy requirements of all classes of its users. These users can be categorised crudely as 'security manager' and 'incident reporter' and 'incident analyst'. Each of these categories has its own needs in terms of scope and means of access. All will have a requirement in relation to the three standard three aspects of security infrastructure; namely: Confidentiality; Integrity, and Availability (Sandhu et al 1991) and (Jajodia 1996).

Confidentiality is vital for any incident database. Due to the nature of its data, such a database could be an invaluable tool for all kinds of hacker or criminal. Adversaries might be able to identify precisely the hardware and software that a company uses. The database might be used as an on-line hacking tutorial to advice adversaries on how to break into specific systems. The problem becomes more apparent if the database is going to be accessed via the Web. In order to trust the database, there must be security procedures in place to protect data stored in it.

Currently most CSIRTs use the World Wide Web as the medium to provide access to their databases. Due to the requirements of security in the WEB context, CSIRT's provide users with only a narrow view of the details that the incident includes. Usually this deals with the low level technical details of the attacks used and countermeasures for the attack. These details omit relationships to particular features within the target system. Although helpful to the technical expert they are far from useful to the manager of a company. Corporate managers need to identify the managerial information related to an incident. Examples of such are: an average cost to the enterprise; the time the company needed to recover from the incident, and, statistical data. The statistics will relate to the frequency of the attack and/or the type of companies this incident targets. This information will assist managers in identifying potential 'need to secure' points and allow them to calculate a budget and/or extend the organisation's *high level* security policy. For the security manager it would not be helpful to display a huge amount of technical information on the screen when data incidents are retrieved. The need for security limits the utility of the current systems.

For current systems the retrieval of information takes place through the use of a simple search engine, using the name or code for the incident as the search key. The current data retrieval engines used by CSIRT's do not allow users to build their own *smart* queries. Security experts and security managers requiring access to the database should not need to become expert in the structure of the security incident database.

The above requirements of security managers raise issues in relation to the operation of a CSIRT database as well as issues for the scope of the data model for the database structure. In addition to having a data model that provides sufficient scope, the way the content is accessed must be appropriate to the user. Examples of smart queries that would be useful to both a security expert and a manager include:

- How many security incidents involved an internal user?
- How many security incidents targeted a buffer overflow in the Apache Web Server?

As noted above the answers for the different category of user might need to be different. The software behind the database should create views of the data targeting different classes of user. Example views could include: The *Management view,* and, the *Technical view*.

The above discussion has focussed on issues associated with retrieval. The key requirements of security and the incident reporting process force the CSIRT's to adopt manual or a semi-automated mechanism for capturing incident descriptions. To enhance security CSIRT's use techniques such as telephone, Fax or emails to gather information of security incidents. These can make the process of registering an incident slow and also make technically oriented personnel uncomfortable. An example of this can be seen in (CERT 2000) , where the form used by the CERT organisation is shown. It allows companies to register their computer security incidents, using FAX or email.

The Cobra Model

The *Object Management Group* (OMG) introduced *Common Object Request Broker Architecture* (CORBA) in 1990. The CORBA model (fig.1 (OMG 1998c)) was aimed at providing an environment where software products, from different vendors could work together (OMG 1998c). In CORBA environment software products can exchange data and processes regardless their manufacturer, their operating system and the way they have been built.

Although CORBA aims at providing a standardised object oriented environment the programs that use this model do not have to be fully object oriented (i.e. Pseudo objects can be created for non-object software implementations).

CORBA does not provide ways of implementing a software package but introduces a way of creating an interface so other programs can access it. In order to build such an interface CORBA introduces an interface language called *Interface Definition Language* (IDL).

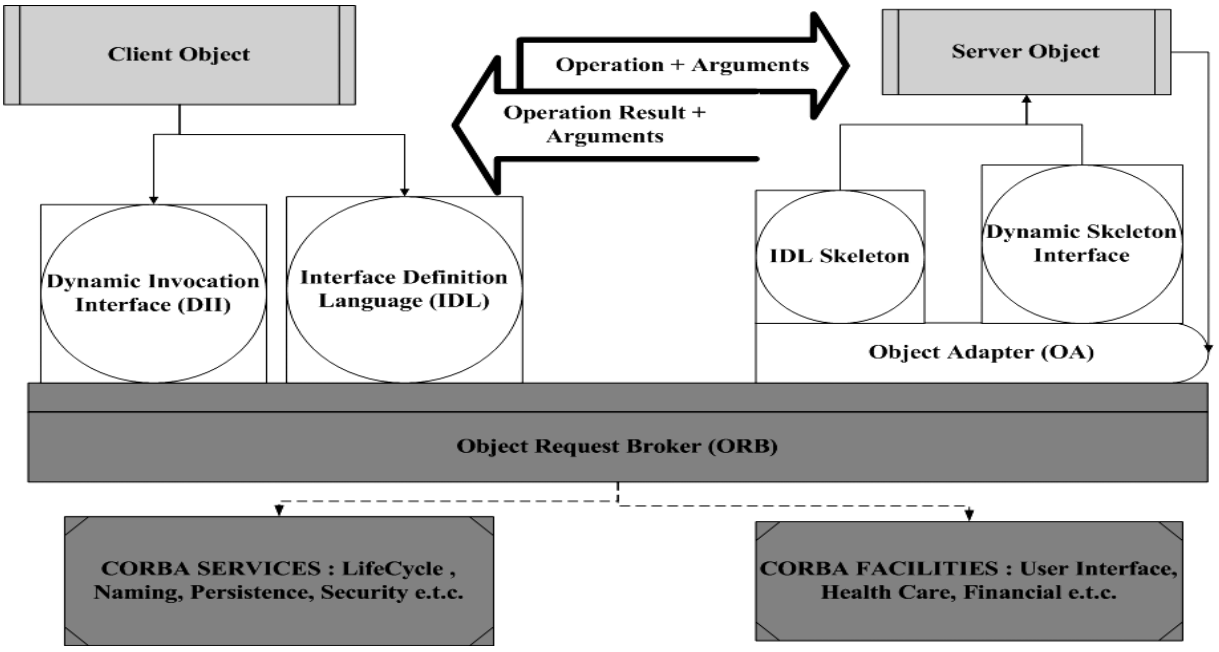

Figure 1: The CORBA Model.

The CORBA/Java combination is the heart of the OMG's *Object Vision* (Orfali 1997). This combination allows the development of smart applications, that can locate the object services they need anywhere around the Internet without considering issues such as O/S, platform or implementation language. The applications of this type are based on component architecture. This allows software vendors to implement, modify or remove components from an application with a great deal of flexibility. Java applets can use the CORBA's *Dynamic Invocation Interface* to identify new server objects and generate server requests "on-the-fly". The session between the Java applet and the CORBA server objects will persist until either side decides to disconnect.

Current research (Blackbird et al 1996) uses the common gateway interfaces (CGI) to allow clients to access and invoke CORBA objects located on the WEB Server machine. To enhance interoperability of the data structures the model uses, OMG have tried to incorporate XML into the model (Schmidt 2001). XML is used in the description of the server objects as well as to describe the information servers and clients are exchanging.

To fulfil the security requirements that a distributed system has CORBA provides a description of a Security Service. The Security Specification establishes the *Security Objects'* needed functionality along with the relations between them. The document in (OMG 1997) describes the functionality that a CORBA secure implementation must provide. By using this as a framework and following the guidelines included in the specification the CORBA

vendors can develop security services that not only can provide adequate security but are going to be able to intercommunicate. Currently a number of known security protocols have been used to provide secure communications in a CORBA environment. Examples of these include Kerberos, Sesame and SSL.

Proposed System

Currently a number of research publications have proposed the use of the CORBA model to access web based databases (Bouguettaya et al 1999a and 1999b) and (Killic et al 1995). The system briefly discussed here uses the CORBA model to allow registered organisations to fully access a CSIRT database. The new system allows access through the TCP/IP protocol. In addition to this the new system automates the process of recording an incident by providing the ability to implement the client of the system as part of the overall company's security management console.

Every organisation that wishes to acquire help with its security incidents will need to pre-register with the CSIRT. The first time a registered security administrator enters the CSIRT Web Site he/she will authenticate himself/herself to the server with the use of an X.509 digital certificate. The user will then download a java applet. After setting up the applet, the applet will open a connection to the CSIRT database server. From there the user will be able to update his/her company's incident record and/or perform queries to the database (Fig. 2). To allow more security the digital certificate will contain the privileges that this user has over the database.

To provide friendly access a Natural Language Interface Database (NLIDB) system e.g. (Androutsopoulos et al 1995) and (Ott 1992) is used. These systems provide the ability to use regular English expressions to search the database instead of SQL queries. Such a system is responsible for translating the English expression into an appropriate SQL statement and for formatting the search output into an acceptable form. Initial designs for the MS SQL Server and MS English Query have been constructed and the examples given in section 3 of this paper have been shown to be relevant to the proposed architecture.

The NLIDB server can create different views depending on the type of the user (i.e. manager or technical personnel). This could add a second layer of security to the system due to the fact that we can program the server to hide the result from 'confidential' fields.

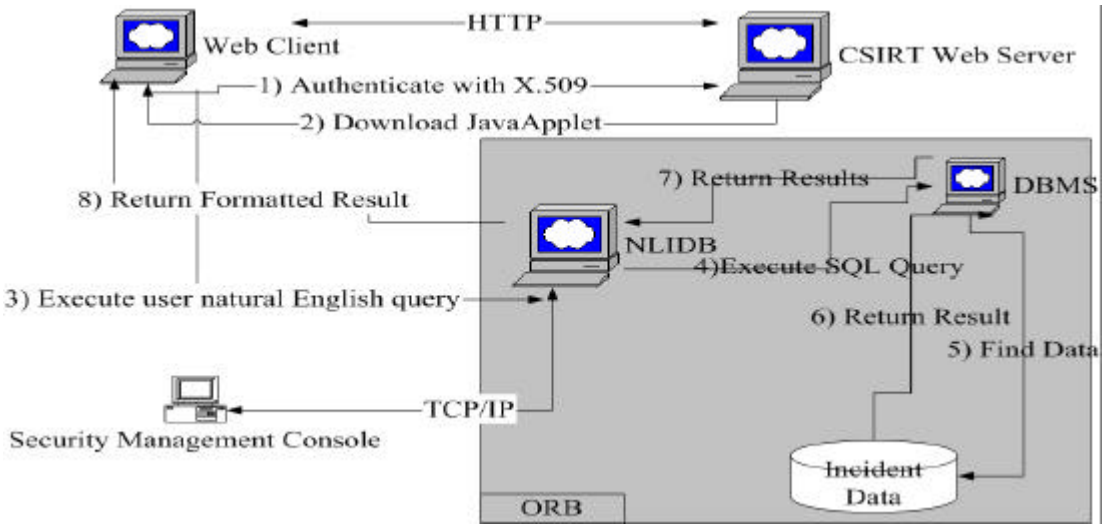

Figure 2: Proposed System

Another advantage of this model is that it allows interconnection with existing security management consoles. Due to the open architecture that the CORBA model provides, security vendors are able to incorporate functions that will allow their product to access the database without the use of the World Wide Web(WWW). This will enable security managers to maintain distributed records of their systems security history. This provides security systems with the ability to register automatically a security incident as soon as it happens (i.e. intrusion detection sensors can record an anomaly as soon as they detect one). In addition the CSIRT could update the intrusion detection sensors and firewalls of the registered companies to detect and stop the new anomaly. To be able to provide this functionality there are a number of problems that need to be solved first. Examples are: deciding on a common incident structure, and, ensuring that the detected incidents are not false.

The client software of the proposed system, either in the form of java applets or embedded code in the security management consoles, will be able to use CORBA's *DII* to identify and locate new services. CORBA will allow CSIRTs to add new functionality on demand.
An additional advantage of the proposed system is that although CSIRTs may use different data structures to store security incidents their DBMS systems will be able to interconnect and exchange information (Fig. 3). By interconnecting they will provide registered companies with a network of interconnected databases. This will provide a better variety of information that will enhance the awareness of companies' employees and the efficient production of security related statistical information.

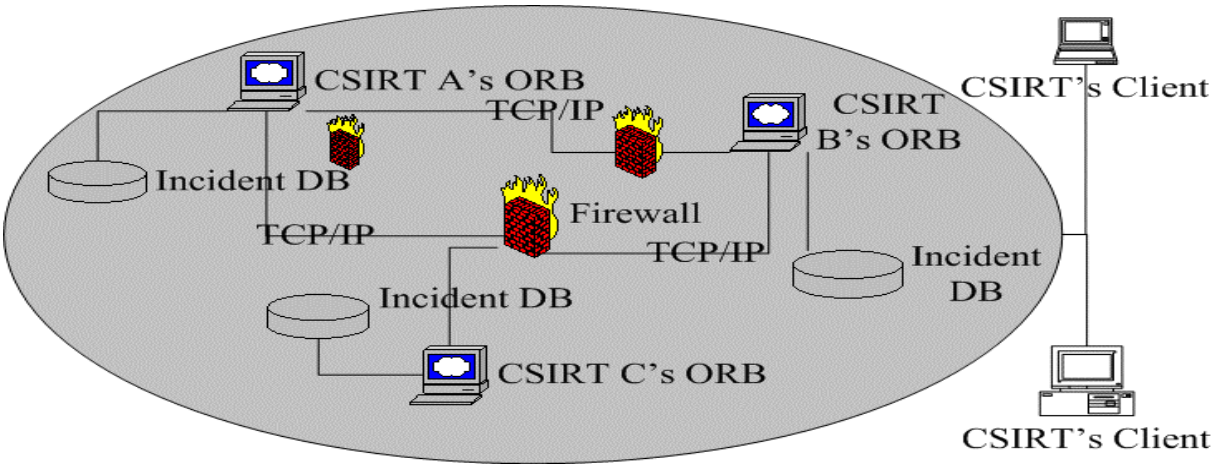

Figure 3: Interconnecting CSIRT

Conclusions

IT attacks are currently on the increase. They are a significant drawback to the Internet's evolution. A Computer Security Incident Response Team is one of the best weapons that the IT community possesses against cyber terrorism. They promote not only security awareness to the IT community but assist companies in tracing evidence that will aid in the arrest of the attackers. In addition to that they provide a repository of information on attack techniques and their countermeasures. This information is analysed by the team to provide information on new trends in attacking as well as patterns that can be used to detect future attacks.

Developing a CSIRT is not an easy task. It includes both technical and managerial issues that have to be resolved before the team is ready to accept and assist in any security incidents.

This paper has concentrated on describing two of the technical issues: Incident data structure, data Acquisition, and, data accessibility.

The system briefly described in this paper uses the CORBA model to allow an automatic registration and acquisition of security incident related information to take place. Additionally, the system allows the creation and use of smart queries by the CSIRT's personnel as well as by the client organisations. The use of the CORBA model provides the effectiveness of implementing the system along with the security management console that organisations use today and/or as part of the CSIRT web site. CORBA implementations provide a security service that is adequate in fulfilling the security requirements that the proposed system has. The incident database structure to be used must ensure that private and public information are stored in separated files having the private ones encrypted at all times. CSIRTs that implement the proposed model will allow more automatic interoperability to take place between them. This will promote even more security awareness and collaboration.

References

Anderson R., (1994) *Why Cryptosystems fail*, Communications of the ACM, **37**:32-40.

Androutsopoulos I., Ritchie G.D., and Thanisch P. (1995) *Natural Language Interfaces to DBs- An Introduction*, Natural Language Engineering, Cambridge University Press, **1**,1:29-81.

Belsis A. M., Godwin N., Smalov L. (2002) A Security Incident Data Model, *Proceedings of the 17th International Conference on Information Security*, Egypt.pp.485-493.

Blackbird P., Gransart C., Geib J.,(1996) *CorbaWeb*: In WWW and CORBA Worlds Integration. *Advanced Topics Workshop on Distributed Object Computing one the Internet, in conjunction with 2nd Conference one Object-Oriented Technologies and Systems (COOTS)*, Toronto, Canada.

Bouguettaya A., Benatallah B., Hendra L., Beard J. and Smith K. and Ouzzani M.(1999) World Wide Database- Integrating the Web, CORBA and Databases. *Proceedings of the SIGMOD Conference*, pp. 594-596.

Bouguettaya A., Benatallah B., Ouzzani M. and Hendra L., (1999) Using Java and CORBA for Implementing Internet Databases. *Proceedings of the 15th International Conference on Data Engineering*, pp.218-227.

Brownlee N. (1998) *Expectations for Computer Security Incident Response*, RFC 2350, Available from World Wide Web: www.ietf.org/rfc/rfc2350.txt.

CERT (2000) *Incident Reporting Form*. Available from World Wide Web: www.cert.org

Commission of the European Communities Security Investigations Projects, (1992) *Project S2003-Incident Reporting a European Structure "Final Feasibility and Strategy Report"*. Version 1.0, Report No19733.

Debar H., Huang M., Donahoo J. D. , (2000) *Intrusion Detection Exchange Format Data Model (IDEFDM)*, Available from World Wide Web: www.potaroo.net/ietf/ids-wg-idwg.html.

Demchenko Y. (2001) *Incident Object Description and Exchange Format Data Model and Extensible Markup Language (XML)*, Internet Draft, Available from World Wide Web: www.ietf.org/ids.by.wg/inch.html

Icove D., Seger K. and VonStorch W. (1995) *Computer Crime: A Crimefighter's Handbook*, O'Reilly Inc.

Jajodia S., (1996) *Database Security and Privacy*, ACM Computing Surveys, **28**,1.

Killic E., Ozhan G., Dengi C., Kesim N., Koksal P. and Dogac A. (1995) Experiences in using CORBA for a Multidatabase Implementation. In: *Proceedings of 6th International Workshop on Database and Expert System Applications*, London.

NIST (1991) *Establishing a computer security incident response capability*. Special publication 800-3. Available from World Wide Web: www.nist.gov.

Object Management Group (OMG) (1997) *The Common Object Request Broker: Security Specification*. Framingham. MA 01701. USA.

Object Management Group (OMG) (1998c) *The Common Object Request Broker: Architecture and Specification*, Revision 2.2. , Framingham. MA 01701. USA.

Orfali, R., Harkey, D., Edwards, J. (1997) *Instant CORBA*, John Wiley and Sons Inc.

Ott N. (1992) Aspects of the automatic Generation of SQL Statements in a Natural Language Query Interface, Information Systems, **17**, 2:147-159.

Power R,(2000) *CSI/ FBI Computer Crime and Security Survey*, Computer Security Issues and Trends, **VI**, 1.

Sandhu S. R. and Jajodia S. (1991) *Honest Database that can keep Secrets*, Proceedings of the 14th NIST-NCSC National Computer Security Conference, Washington D.C., pp. 267-282.

Schmidt C. D. and Vinoski S. (2001) *Object Interconnections: CORBA and XML, Part 1: Versioning*, C/C++ Users Journal, Available from World Wide Web: http://www.cuj.com/experts/1905/vinoski.htm?topic=experts.

West-Brown J. M., Stikvoort D., Kossakowski K. (1998) *Handbook for Computer Security Incident Response Teams (CSIRTs)*, Carnegie Mellon, Software Engineering Institute, Technical Report CMU/SEI-98-HB-001.

OPINION PAPER

Is The IW Paradigm Outdated?
A Discussion of U.S. IW Theory

Timothy L. Thomas

** The views expressed in this report are those of the author and do not necessarily represent the official policy or position of the Department of the Army, Department of Defense, or the U.S. government.

Foreign Military Studies Office
Fort Leavenworth, Kansas
E-mail: thomast@leavenworth.army.mil

Abstract

The term information war (IW) helped describe one aspect of the unfolding revolution in military affairs in the 1990s. Today, technological developments are integrating the data processing capabilities of machines and the mind in ways not possible a decade ago. As a result, the old IW paradigm may no longer be applicable, making other potential paradigms and terms worthy of consideration. As the future unfolds, it will be interesting to see if Pentagon theorists use IW or a new term to express a threat to the security of the country, a category of warfare, a method of defense or influence, or leave the concept alone as a conceptual umbrella for a host of terms. Or, will the Pentagon simply update IW theory, perhaps developing Information Peace or Mind-Machine concepts that complement IW?

Introduction

Many military analysts believe the end of the Cold War signaled the close of the age of conventional weaponry. At this juncture, a switch was made to acquire weaponry based on technological achievements spawned by the so-called revolution in military affairs. This latter concept had been under consideration for several years. As one analyst noted

Future historians might well cite the years 1993 and 1994 as the period during which the US military and associated national defense organizations identified Information Warfare as a conceptual vehicle for transitioning from the precepts of the Cold War into the new global realities of the Information Age. This concept is gaining momentum throughout the national security community at a breakneck pace. (Berkowitz, 1997, pp. 175-190.)

American industry and business were the first organizations to fully embrace the power of information-age technologies, and the military soon followed. The Internet as we know it was a military research project to develop redundant communications paths that would work if some communications nodes were destroyed. The Defense Department learned to utilize these technologies in logistics and administrative endeavors, and created attack and defend weaponry that could process information quicker and with greater precision than could its adversaries. The overwhelming effectiveness of this ability was most decisively demonstrated

ISSN 1445-3312 print/ISSN 1445-3347 online

for the first time during the Gulf War. The coalition victory over Iraq was labeled as the first battlefield reliance on information technologies to achieve victory.

US military theorists attempted to describe this new emphasis on technology under the cover of a joint doctrine, Joint Publication 3-13, "Joint Doctrine for Information Operations." (DoD, 1998). Also discussed in the same volume was information warfare (IW). These concepts were new and as a result loosely defined, not completely mirroring the terms that composed them. Those responsible for defining the terms were stepping into totally unmapped territory, and had no frame of reference from which to work other than the existing doctrines of psychological operations (PSYOP), operational security (OPSEC), military deception, and similar related topics. Their work was admirable. IO and IW were used as metaphors to express the technological transformation that was underway in times of peace and conflict, respectively.

Recently, however, conceptual writers working on IW doctrine have adopted some radical changes. Most important for purposes of this article is that IW has been removed from the Army's information-related terminology. The familiar term "information operations" now serves all functions associated with information age processes and weaponry. Such a radical move indicates that the time is ripe for a review of some of the other concepts associated with information operations as well. As the analysis below demonstrates, some of the vital terms around which IO has been constructed remain loosely defined; some international terms that might support IO theory are totally absent from consideration in the US lexicon; and some new paradigms describing the information age are beginning to appear. This article makes several recommendations to tighten up terminology, especially concepts such as information superiority, and focuses squarely on other potential paradigms for interpreting current and future developments, offering one paradigm for discussion.

Terminology and the IO/IW Paradigm: The Early Decisions

An authoritative definition of IO and IW reasonably should encompass the accepted meanings of three components: information, operations, and war. When the terms in JP 3-13 are examined, however, disconnects become apparent regarding how military analysts defined the terms some ten years ago, and how they could have defined them. The old definitions worked because they were new and no one REALLY understood what the terms meant. After thousands of pages have been written on IO and IW, it is useful to quickly review these definitions, pointing out their deficiencies that were not apparent at the time.

Three authoritative sources are used here to examine the definitions of information, operations, and war. The three sources are Webster's Dictionary, the Department of Defense's Dictionary of Military and Associated Terms (DoD, 2001), and Joint Doctrine for Information Operations (DoD, 1998). The reason for using a source outside of the military realm is obvious upon closer examination: first, one of the three terms (war) under discussion *is not defined in either JP 1-02 or JP 3-13*, and therefore Webster's is necessary. Secondly, Webster's dictionary is a generally accepted source while both JP's contain military related terminology that is not generally accepted by the academic community at large. The JPs have not been reconciled with general sources; rather words are defined from a military paradigm and may not mean what someone outside the military thinks they mean.

The two primary definitions for the word "information" found in Joint Publication 1-02 are facts, data, or instructions in any form," and "the meaning that a human assigns to data by means of the known conventions used in their representation." (DoD, 2001, p. 254) JP 1-02

defines an operation as military action or the carrying out of a strategic, tactical, service, training, or administrative military mission; the process of carrying on combat, including movement, supply, attack, defense and maneuvers needed to gain the objectives of any battle or campaign." (DoD, 2001, p. 384)

It would appear, based on a look at these definitions, that IO could have been defined from a purely military point of view in one of three ways. First, as "a military action using facts, data or instructions." Second, as "carrying out a strategic or tactical military mission using facts, data or instructions." Finally, as "the process of carrying on combat, including attack and defense means, to gain the objectives of any battle or campaign using facts, data, or instructions." Joint Publications 1-02 and 3-13 define IO as "actions taken to affect adversary information and information systems while defending one's own information and information systems." These official definitions mirror the potential definitions suggested above in a tangential manner. The official definition's focus is clearly on the information systems of equipment, and not on mental perception or reaction, a huge component of IO, or on facts, data or instruction. The gamut is insufficient to clearly define IO. The Army's relatively new Field Manual 3.0, Operations, describes the attainment of information superiority (the goal of IO) as capable of putting disparity in the enemy commander's mind between reality and his perception of reality. (Department of the Army, 2001) It thus discusses influencing the mind of the commander, giving more attention to PSYOP than did the old definitions of IO that relate primarily to equipment and give scant reference to the mind.

Similarly, IW could have been defined differently than it was in JP 3-13 and JP 1-02. War is defined by Webster's dictionary as a state of usually open and declared armed hostile conflict between states or nations." (Websters, 1998) JP 1-02 *does not define war*. This is an interesting point. How can one possibly define IW if we do not define "war"? The question is worth asking, for avoidance of controversy is hardly an acceptable price for lack of clarity. This can become a troubling condition. For example, consider the emphasis we put on the correct determination of an "objective" or a "center of gravity" as operational or strategic principles. If we do not define the terms properly, can we make the proper determination? Was a weak definition the reason that IW is no longer in the US army's lexicon? If so, we will need to inform the international military community (coalition forces that support us), because they are still using the IW phraseology.

JP 1-02 defines IW as "information operations conducted during times of crisis or conflict to achieve or promote specific objectives over a specific adversary or adversaries." The latter part of this definition sounds more like the definition of an operation, in that an operation carries on combat to "gain the objectives of any battle or campaign." Further, the definition of war can only be inferred to mean "crisis or conflict" based on the definition of IW in JP 1-02. That is the only difference between IO and IW.

This problem is exacerbated when foreign concepts are bumped up against U.S. concepts. For example, the 1986 *Russian Military Encyclopedia* (SME) defined military information (there was no entry for information) as "information of a military nature, as well as the process of transmission and receiving of such information." (SME, 1986, p. 294) An operation was defined by the encyclopedia as "an aggregate of battles, engagements, strikes and maneuvers, coordinated and interlinked in objective, tasks, place and time, by various force organizations, conducted simultaneously and sequentially according to a common concept and plan, to accomplish missions in a theater of operations, a strategic or operational sector, or within a specified period of time; a form of military operations." (Akhromeev, 1986, pp.

514-515) War was defined as "a sociopolitical phenomenon, continuation of politics by violent means…armed struggle comprises the specific content of war." (Akhromeev, 1986, p.151).

If the definition of war as found in Webster's dictionary (JP 1-02 cannot be used since it does not define war) is combined with the definition of information from JP 1-02, then a sample definition of IW could be "open and declared armed hostile conflict between states or nations using data, facts or instructions in any form." This definition makes little sense. A more appropriate but still imprecise definition would have been "open and declared armed hostile conflict by nations or states that utilizes information or information-based systems and processes to attack human or system processors."

Even more to the point, information does not actually go to war against other information, further casting doubt on the idea of IW. Data or electron streams can be directed against one another to collide or interfere or influence movement, but they are not in open and declared "armed conflict." Electrons might collide with other electrons, laser beams may try to destroy computer chips, and directed energy beams may try to destroy satellites. But this is not "IW." Perhaps it could be called beam confrontation, or electron stream conflict where computer chips and other data-processing elements are the objectives of attack. However, as noted above, this is a mute point. The US army no longer has IW in its lexicon, although it does remain in the digital version of JP 1-02.

Other information-related terms

There are several nations, such as Russia and China that define other information-related issues. These include terms such as information weapons and information-psychological actions. JP 1-02 and JP 3-13 do not define an information weapon, just as they do not define war. There appears to be a long-term unwritten policy in military circles in the US not to define an information weapon. Yet much of today's weaponry is loaded with computer chips and other information technology, the cornerstones of the information age. The Russians define information weapons in great depth and specificity. They ask "how can you have an information war if you do not have information weapons?" Can we have tank warfare without tanks? This term and others might be worthy of future consideration by IO specialists in this country. One of the greatest strengths of the US armed forces is its ability to learn from other armed forces (for example, the US studied and then adapted the term operational art from Soviet theoreticians). Perhaps now is the time to data-mine foreign IO theory for some of the good ideas that they have developed, and see if they are applicable to the US paradigm.

For example, it is interesting to ask, some ten years down the road, why did US theorists, at the height of peace operations theory at the end of the Cold War, develop the concept of IW instead of "information peace"? That is, how nations might use information technology to prevent conflict. Why did we choose information war? The Soviet threat had evaporated, and a worldwide scare generated by a new term, IW, was the last thing Russian reformers needed. Many in Russia interpreted the term as a method of mind control. Now, however, the time appears right for cogent arguments to be advanced to reject or modify some terms and concepts, and to promote new conceptual vehicles.

And there are terms in the US IO lexicon that could stand a scrub. One of the most important is information superiority (IS). JP 1-02 defines IS as "that degree of dominance in the information domain which permits the conduct of operations without effective opposition."

(DoD, 2001, p. 255). JP 3-13 defines IS as "the capability to collect, process and disseminate an uninterrupted flow of information while exploiting or denying an adversary's ability to do the same". (JP 3-13, 1998) Most recently The Department of the Army (2001, p11.2) defined information superiority as "the operational advantage derived from the ability to collect, process and disseminate an uninterrupted flow of information while exploiting or denying an adversary's ability to do the same."(Department of the Army, 2001) If anyone reading this or any other article collects, processes, and disseminates the information therein, they do not necessarily have information superiority. What if all of the information in this essay is misinformation? That does not provide information superiority, only information inferiority. To process is not to analyze. Or is it? One argument is that "process" is part of the intelligence cycle: converting raw data to a form that is usable by an analyst or for immediate action by a commander. That is, it hasn't been analyzed yet. It is only "useable" by an analyst. Another argument would indicate that analyze might be included in the term "process." The word "information-based processes" in JP 3-13 includes the term analysis ("processes that collect, analyze, and disseminate information using any medium or form"), making it appear that JP 3-13's authors do not make as big a distinction between process and analysis as might be implied. Perhaps too big a deal is being made of this slight oversight, but it would be nice to add the "reminder" word analysis to the IS definition.

The reason for such reminders is that processing and not analyzing information caused a huge waste of munitions during the air war over Kosovo as fighter pilots targeted mockups that appeared to be real targets. How else can one ascribe the difference in the number of tanks we thought we had "killed" in July of 1999 (according to General Wesley Clark, 110) and the final tallies, some of which ranged as low as 26 tanks? And this occurred in the face of near total information superiority when no air force was flying against ours and we owned the airwaves! The US had "self-deceiving information superiority" as a result. FM 3.0 describes information superiority as "when commanders synchronize all three contributors (intelligence/ surveillance/ reconnaissance, information management, and information operations) and it is greater than the enemy's." (Department of the Army, p11-2, 2001) This descriptor considers that information has been processed and analyzed, one hopes, in order to attain information superiority.

One is left with the feeling that IW and some other information-related terms do not exist in a pure state but are simply metaphors for expressing aspects of war using high technology weapons or computers, and to a lesser extent terms associated with psychological operations. But terminology is not the only concern with the old model that has served us so well for ten years.

IO applies to no specific or conceptual model (conventional, non-traditional, etc.) but instead conjures up a unique mental model composed of many elements that exist independent from other forms of warfare. FM 3.0, Operations (Department of the Army, 2001) lists ten different "elements" of IO: military deception; counter deception; operations security; electronic warfare (e-attack, protection, support); information assurance; physical destruction; psychological operations; counterpropaganda; counterintelligence; and computer network attack and defense. Such a broad definition makes it hard to distinguish what IO is not, not what it is! Further, if IO's goal is to produce a disparity in the enemy commander's mind between reality and the perception of reality in order to disrupt the ability to exercise C2, as FM 3.0 notes, then IO is really nothing new. Deception has been doing this for years. The methods are the same, but the means (sensors, satellites, holograms, etc.) are different as well as the precision and speed of destruction involved. But IO is something new. Today

weaponry does more than just disrupt the ability to conduct C2 via deception or PSYOP. Now weaponry can shut down the data processors of both weapons (computer chips) and the mind (neurons). There is more at stake than just deception.

In its purest form, attacks with electrons or leaflets or other means attack equipment and weaponry on the one hand, but also can attack the logic and decision-making capability of the commander on the other, according to Army FM 3.0. That is, there is a huge psychological warfare component of IO, much greater than is generally implied or discussed. There are information operations against the brain on a daily basis in nations worldwide, especially in advertising and the mass media, where the goal is to influence or persuade through pure influence, debates and tests of logic. Less considered is the more sinister form of an information operation against the brain that uses acoustics or other devices to shut down the normal processing of the brain, much like a laser tries to destroy computer chips. U.S. specialists tend to put these operations out of the IO fold and into a field known as non-lethal operations. Most definitions of PSYOP are all about the first, or soft, use of IO: to influence the logic in someone else's head (leaflets, loudspeakers, deception, etc.), or to use counterpropaganda as an element of defensive PSYOP. Soft PSYOP, however, is not the same as protecting the neurons in your head from being fried by a non-lethal device. Data attacks a human's logic in a soft IO PSYOP attack, and electronic or non-lethal streams of data attack neurons in the brain in a hard IO PSYOP attack. Lasers do the same to computer chips, but the latter is more easily identifiable with IO (physical destruction). What about mental destruction?

In summary, the theories of IO and IW have served the US well for the past ten years. They have given us a conceptual model through which to understand the changes that new technologies have brought us. This paradigm made sense ten years ago. But already, new models and concepts are emerging: Network-Centric War, effects-based strategies, asymmetric war, and operational prototyping are but a few of them. Some of these, based on the discussion above, most likely are already deficient since we have no definition of war! Others are being subsumed under the IO mantle.

Yet another model or paradigm, human and equipment data processors, is offered now for your consideration. It attempts to capture much of the discussion above in a simpler form and under an old term, data processing.

"Data processing Operations:" Another way to think about the Information Age

The manipulation of data (information) has played an important role throughout the history of armed warfare. Before the creation of the computer chip (a data-processor), an information operation meant influencing or manipulating the actions of the decision-maker, the human. It was more of a psychological operation than an information operation, since there was a heavy reliance on intimidation and deception. In reality, the attack was on a human's logic or on the emotion of fear. Just as today, data (leaflets, messages, newspapers, etc.) or activities (a show of force, atrocities, etc.) were the means to manipulate the data processor known as the mind via deception or intimidation. The transmission of data was slow in ancient times, and this also affected the manner in which data was analyzed.

Today, data processors in weapon systems have provided for a vast improvement in the acquisition and transmission of information, ensuring quick and precise attacks on targets even from standoff positions. Data processors allow commanders to mass effects quickly and

precisely at decisive points across broad geographical areas. However, feeding false information into the data processor can still fool the logic of even a computer chip. Wrong information in, wrong information out. A computer chip does not have the ability to "fear" data, but can be programmed to reject certain kinds of data.

Data processors form the core element, the heart if you will, of sensors, satellites, and computers. Thus, computer network operations (CNA, CND, and more recently computer network exploitation or CNE), or attacks on sensors or satellites, are in reality attacks against data processors. In like manner, psychological operations, deception, and even non-lethal operations are directed against the data processor known as the mind. PSYOP and deception have reached new levels of maturity, in that holograms, morphed images, and other virtual representations of reality now have the potential to influence people like leaflets and loudspeakers once did. Non-lethal weaponry, such as acoustics or stun guns, is capable of momentarily shutting down the data processor known as the brain. Non-lethals can be described as soft influence means (leaflets, soft attacks) or incapacitating and even debilitating (hard attack) means. Unfortunately, this important latter attack method is not covered by present day IO theory in the US, which is focused on systems and equipment. 'Non-lethals' are in a separate category for analysis.

People have ignored the fact that the mind has no firewall for too long (although any non-lethal specialist would argue this fact—here IO specialists are addressed!). The primary emphasis on networks and pieces of equipment missed the most exposed computer/data-processor on the battlefield, the human head. PSYOP and military deception are the only elements of those ascribed to IO that are concerned with the human information security feature, logic. A non-lethal substance is much more insidious—it attempts to alter or destroy the functioning of the brain's neurons just as an electron stream or laser beam attacks the data-processor known as the computer chip.

Further, in a recent interview with Wired magazine, Mr. Andrew Marshall of the Net Assessment Office of the Pentagon, often referred to by Pentagon insiders as Yoda, underscored the importance of the mind and its implications for future warfare scenarios. He noted that

People who are connected with neural pharmacology tell me that new classes of drugs will be available relatively shortly, certainly within the decade. These drugs are just like natural chemicals inside people, only with behavior-modifying and performance-enhancing characteristics. McGray (2003, p.117) joked that a future intelligence problem is going to be knowing what drugs the other guys are on.

Thus the data processor, possessed by both equipment (computer chips) and humans (neurons), is the actual center of gravity of future attacks. It might be soft attacks on logic or hard attacks on chips or neurons. The old IO/IW paradigm did not focus on the data processor as the objective of an attack, but rather offered elements and other descriptive criteria to describe IO. However, much of the problem with data processors is related to our reluctance to view a human as a data processor. We are not accustomed to doing so, and don't feel comfortable putting non-lethals into the IO lexicon as a result. Physical destruction appears to have a comfort level of acceptance, but mental destruction does not.

Further, the human is always the interface between the input and output of data processing. The mind must not only ward off deception and acoustic attacks against it, it must also

interpret what is downloaded from a computer or satellite, and what is acquired by counterintelligence means (for example, counter EW or HUMINT operations) or developed via counterpropaganda operations. Equipment can produce false outputs that the human interface must be coy enough to process, analyze, and interpret. Once again, the focus is on a human's ability to use proper logic to come to the correct conclusion. The computer-operator or machine-mind interface is one of those centers of gravity for the technological age that people seldom mention. The journal *Technology Review* recently wrote about this development, noting that efforts to link brains and computers could result in thought-controlled robots, enhanced perception and communications, and might make you smarter. (Technology Review, 2003)

Websters (1988) dictionary defines data-processing as the converting of raw data to machine-readable form and its subsequent processing (as storing, updating, combining, rearranging, or printing out) by a computer." The online *American Heritage Dictionary* defines a data processor as a device that performs operations on data; a machine for performing calculations automatically; or a person who processes data. JP 1-02 does not define data processing. This is not surprising since it is not a military specific term, although military forces in almost every aspect of their day-to-day lives (like civilians everywhere) and wartime activities use its capabilities.

Knocking out or manipulating the organizer and distributor of data, the data processor, is the focus of the new paradigm. The data processor is the objective. As a result, it would be more correct to position data-processor wars at the top of the hierarchy of the concept and to position information as a sub-element, a means to influence the data processor. *Attacks on computers or the mind, whether electronic, laser, or other, are designed not to attack information but rather to disable, manipulate or destroy the data processor*. In the case of a sensor, satellite, electronic warfare platform, or a computer, it is an attack on 1's and 0's of computer-based language, or on the computer chip itself. With regards to humans, special light or TV frequencies that induce photoelectric epilepsy, or other forms of debilitating light that attack the actual functioning of a human's data processor, the brain, are the areas of concern. Finally, regarding persuasion management or deception activities, these can be used against either equipment or a human.

If data-processing operations were broken down into two categories, equipment and the brain, what shape would the categories take? If one were to look at the elements listed in FM 3.0, psychological operations, military deception, counter deception, counterpropaganda, and counterintelligence would be listed under the brain as elements designed to influence this data-processor. Non-lethals, not one of the elements in FM 3.0 (Department of the Army, 2001) would also have to be added to the list, since they can shut down both logic and bodily functions. With regards to equipment, electronic warfare, information assurance, and computer network operations (CNA, CND, and CNE) would be listed. Of course, military deception is not purely a function of the human side of the equation. Military deception could be used against a piece of equipment's data processor just as easily as it could be used against a human. A sensor that is fed false signals is one example. Physical destruction and operations security also could fit both the equipment and brain categories.

Are there other paradigms besides data-processing? Of course there are, and several were already listed. The one offered here is just a simple example, and far from the most creative. Most paradigms are susceptible to the same problems as IO and IW, however. They have to be properly defined, meaning they should not offend more traditionally accepted definitions

but adapt to them. And care must be taken not to insist on mutually exclusive categories, as IO and IW do. Another suggestion is to divide the conceptual model to describe technological change into three parts: kinetic warfare, electro-magnetic spectrum energy warfare, and PSYWAR (or influence war). Each would have specific sub elements (deception, operational security, etc.). Or one could look at the technological revolution as simply dividing the pie into technology and psychological sectors. In any event, much conceptual thought is needed. Data processing is only a way to think about the problem, not a way to define it. And particularly with regard to non-lethals, there are a whole host of international laws that would affect the development of any mind-related concept.

Conclusions

The rapid pace of development in today's technology sphere indicates that the concept of "information warfare" that served a purpose for ten years is now somewhat dated, or at least in need of updating. IO begins before a crisis or conflict begins and is ongoing during the conflict. To say that military deception, OPSEC, PSYOP (look at the US use of leaflets and TV/radio today to influence the Iraqi population and soldiers long before actual conflict), and other elements of IO only occur during conflict is missing the point. It will be interesting to see if Pentagon theorists use IO to express a threat to the security of the country, a category of warfare, a method of defense, or leave it as the same conceptual umbrella for a host of operations, the function it served in the past. Or will a new prism of analysis replace IO itself?

Elements of the old metaphor are still applicable but a new conceptual model is needed, and especially one with more focus on the mind. The old concepts of IO/IW looked almost exclusively at equipment and systems, and gave scant notice to the mind except for soft PSYOP—counter deception, counter propaganda, military deception, and operational security. But, as pointed out in the text above, the focus was purely on how to influence the behavior and opinions of others, which does not concentrate on protecting the mind from either information that could influence behavior or attitudes, or from weaponry that could upset the functioning of the brain's neurons. One way to prevent such attacks from ever occurring is, again, to ensure international laws are in place to thwart such potential applications.

To underscore future developments regarding the mind, one needs look no further than the August edition of *Newsweek*. (Newsweek, 2002) Inventor Woody Norris indicated that a device he created could now put words or images in your head from 100 yards away. The military and law enforcement officials are closely monitoring the results of this experimentation. This development, if it turns out to work, would further emphasize the importance of protecting the mind. Just as data can be fed into a computer, Norris's invention would indicate that data can be fed into a person's mind as well. The *New York Times*, *Popular Science*, and *Business Week* followed up with stories on Norris's invention a few months after the *Newsweek* article.

A clear future problem will be not only validating concepts in the US, but attempting to get nations across the globe to find a common language if/when the problem of IO/IW is brought before the United Nations. Just because we "invented" IW and the Army has now discarded it does not necessarily mean that other nations will do the same. One Chinese officer, for example, noted that IW is ongoing all the time, and IO only happens in wartime. A key Russian concept, around which that country has developed a doctrine and policy, is "information security." Both nations define information weapons. All of these approaches are

not explored fully in current US thinking. Of course, US policy makers have consciously decided to ignore some of these concepts for national security reasons, and their concerns are real.

It is unfortunate that the focus of defense departments worldwide is so focused on IO or IW and not on "virtual peacemaking" or "information peace" concepts. Information technologies were used extensively to keep the sides from fighting after US forces entered Bosnia, and a precedent was set. The presidents of Serbia, Croatia, and Bosnia-Herzegovina were influenced strongly by the use of information technology to come to a common agreement when their borders were divided. At Dayton, the presidents were placed in a room and shown a virtual flyover of crucial border regions. From information in the video, they were able to discuss and develop among them solutions to sensitive issues. Such use must be at the forefront of our efforts, not the afterthought.

In Iraq, information technologies were used to ascertain if the US and other nations would eventually go to war. No one is talking about the UN's use of high technology as a "virtual peacemaking" or "information peace enforcement" operation. Rather, people only are thinking of IO use in a conflict. More attention needs to be focused on the persuasive influence of high technology developments, and not just by the Pentagon. The State Department should be in the lead on this issue. They have developed a good grasp of virtual diplomacy issues, and need to continue to push such agendas and methods for resolving issues via negotiations.

One conceptual paradigm offered here was the data-processing model. And that was all it was, a potential concept. Others are certainly available and should be analyzed. IO, as a result, may be the victim of the next mental scrub over future developments. Whatever model is chosen, it must be simplistic, reflect reality and offer a way for thinking about the evolving nature of the world around us. IO and IW were excellent starters that helped military people understand the changing nature of technology. Now might be the time to move on to other concepts, or to slightly alter the initial IO and IW concepts. Contemporary developments will soon mimic some of the futuristic scenarios that Hollywood filmmakers are portraying (*Minority Report* comes to mind immediately)—and we will need a paradigm from which to understand them.

References

Akhromeev,S.F (1986) *Russian Military Encyclopedia*, Second Edition, Moscow.

Berkowitz, B.(1997) Warfare in the Information Age. *In Athena's Camp*, eds: J.Arquilla, D. Ronfeldt, National Defense Research Institute, RAND, Santa Monica.

Department of the Army (2001) *FM 3.0, Operations*, Headquarters, Department of the Army, U.S. Army Printing Agency, Washington, D.C.

DoD (2001) *Joint Publication 1-02: Department of Defense Dictionary of Military and Associated Terms*, 21 April 2001. URL: http://www.dtic.mil/doctrine/jel/new_pubs/jp1_02.pdf .Accessed on 7 May 2003.

DoD (1998) *Joint Publication 3-13: Joint Doctrine for Information Operations*, 8 October 1998

Huang, G. (2003) Mind-Machine Merger, *Technology Review*, May 2003.

McGray, D. (2003) The Marshall Plan. *Wired Magazine*, Issue 02/2003

Reno, J., Croal N. (2002 Hearing is Believing, *Newsweek*, 5 August 2002, Internet version.

Websters (1998) *Merriam-Webster's Collegiate Dictionary*, Tenth Edition, Merriam-Webster, Incorporated, Springfield, Massachusetts, U.S.A., , p. 599.