CHAPTER

Security Projects for Systems and Networking Professionals

7

Leonidas Deligiannidis[1], Charlie Wiseman[1], Mira Yun[1], and Hamid R. Arabnia[2]
[1]*Wentworth Institute of Technology, Boston, MA, USA*
[2]*University of Georgia, Athens, GA, USA*

INFORMATION IN THIS CHAPTER

- Overview of encryption techniques
- Example projects utilizing encryption technologies
- Overview of wireless security
- Towards the development of a Cyber Security Management framework
- Example of subverting wireless security

INTRODUCTION

Despite the growing demand for security professionals in all areas of computing and information technology, relatively few schools offer degree programs that incorporate hands-on security training. The result is that most security-oriented positions are filled either by security veterans or by fresh graduates who have little or no working security knowledge. A small number of free online resources are available (e.g., http://opensecuritytraining.info), but most security education ultimately happens on the job and only as needed.

This chapter provides an overview of introductory systems and network security by way of several projects and demonstrations. Clearly, this one chapter cannot serve as a complete picture of the vast field of security. Rather, it aims to be a starting point for newcomers to the security field who want to bridge the gap between concept and practice.

The materials presented in this work are derived from undergraduate courses taught in the Department of Computer Science and Networking at Wentworth Institute of Technology (WIT). That department offers two degree programs: one in computer science and one in computer networking. While the computer science degree program is a typical, traditional CS program, the computer networking program is more hands-on and exposes students to a broad skill-set in networking, computer science, and management.

One of the courses now offered regularly at WIT is a cryptography and network security course. The book used in the class is *Cryptography and Network Security: Principles and Practice*, by

Emerging Trends in ICT Security. DOI: http://dx.doi.org/10.1016/B978-0-12-411474-6.00007-4

William Stallings, Prentice Hall. This is an excellent book and is highly recommended to anyone interested in learning more about network security. There are several programming assignments that cover from the ground up symmetric encryption, digital signatures, digital certificates, and RSA public key cryptography. More recently, wireless network security has been incorporated into the curricula at WIT in several different courses, including the network security course, cryptography and network security, and wireless networking courses.

This chapter describes and discusses a selection of the projects, demonstrations, and future materials from these courses.

Background

It is suggested that hands-on, investigative teaching with associated exercises improves the learning performance of students [1]. It is also shown that engaging students in hands-on exercises promotes active learning and helps students develop critical thinking skills [2,3] better than simply covering lecture material in class and leaving students with many unanswered questions. This improves their learning performance and, as a result, increases the likelihood of programs to retain their students [2–9]. A three-year study was performed to determine why Science, Technology, Engineering, and Mathematics (STEM) majors switch to non-STEM majors [10]. They found that students switch majors because of lack of interest, teaching methodology ineffectiveness, and because they feel overwhelmed with the curriculum demands. Successful completion of the introductory courses for the first year is crucial in retaining students in a program, and most lecture courses are notoriously ineffective in engaging students [11].

DefEx is a set of hands-on cyber-defense exercises that focuses on undergraduate development through understanding and problem solving related to security [12]. These exercises include code and system level hardening, problem detection, digital forensics, wireless access point security, cross-site scripting, command and SQL injection, file uploading, and a wireless access point treasure hunt game based on wardriving that requires students to utilize all their skills from throughout the course.

Cryptography

In the cryptography course, the students are given two exams during the semester and a final at the end of the semester. The students are also required to attend every lecture and complete six assignments. Extra credit assignments are provided for students who are interested in learning more. In the next section we describe the major programming assignments and demonstrations we perform to motivate our students.

Assignment in symmetric encryption

The Data Encryption Standard (DES) algorithm used to be the most widely used symmetric cryptosystem in the world. It is a block cipher that was selected by the National Bureau of

Standards as an official Federal Information Processing Standard for the United States in 1976. DES is now considered insecure and has already been superseded by the Advanced Encryption Standard (AES) [13].

The DES algorithm takes a 64-bit plaintext block and a 64-bit-long secret key and transforms them into a 64-bit-long ciphertext block. Decryption must be performed using the same key as was used for encryption and the same algorithm in reverse to reproduce the original plaintext block.

The students learn about symmetric encryption algorithms, and they complete one assignment where they use DES to encrypt and decrypt a memory-mapped file. In another assignment, the students compare the performance of DES and AES by plotting how fast different-size files are encrypted and decrypted.

Assignment in hash functions

Cryptographic hash functions play an important role in modern communication technology. The input to a hash function is a file or stream of any size and the output is a fixed-size digital representation of the file that is normally less than 1KB and serves as the fingerprint of the original file (often called the message digest). It is impossible to reconstruct the original file with only the fingerprint. Moreover, changing a single bit of information in the input would result in a significantly different fingerprint. These algorithms are designed to avoid collision. In other words, it is very unlikely for two messages, *M* and *M'*, to produce the same fingerprint using the cryptographic hash function *H*: *H(M) != H(M')*. Many cryptographic hash functions are based on the so-called MD4 algorithm initially proposed in [14], and they have received the greatest attention.

Students write a program to compute the message digest given different input streams. Then they modify the input in order to produce substantially different digests. The students are challenged to find two inputs that produce the same message digest. We then demonstrate how they can break MD5 using the techniques described in [15,16]. Specifically, we produce two different executable files that have significantly different purposes yet whose MD5 digests are identical. This shows that it is possible to have two different files with the same MD5 message digest and that using MD5 hashing to verify file downloads is not safe.

Extra credit assignment on steganography

Steganography is a technique used to conceal information so that only the communicating parties know about the existence of the information in any form [17]. Steganography uses a cover image, which is an image that has information embedded in it. Then there is the actual information itself, which could be anything from plain ASCII text or another image to pdf files, sound files, etc. After the information has been embedded into the cover image using any of the steganography techniques (such as Least Significant Bit Insertion), a Stego image [18] is obtained. It is also possible to encrypt the information before embedding it, thereby adding another level of security [19].

We demonstrate two applications that use steganography. Both of them are written in Java and are accessible via Web Start technology: HAE (Hide at End) [20] and Stego [21]. HAE

demonstrates how one can hide any document of any size at the end of a PNG picture. The output looks exactly like the input, but the file is bigger in size. This tool relies on the fact that PNG viewers ignore everything after the end-tag of the PNG image. So, right after the end-tag, any other file or information can be appended. Students can use HAE to interactively hide pictures inside other pictures, retrieve them, and check file sizes. HAE inserts metadata (seven bytes) at the end of the first image, which describe the data that it hides after the end of this file. This way, HAE can easily retrieve the hidden information. One can experiment with hiding information after the end of an image by simply running the following command in Windows:

```
copy /B a.gif + p.pdf output.gif
```

This command copies a binary (/B flag), a.gif, appends to it the p.pdf file, and produces the output.gif file. Viewers will open the output.gif file and display the image stored in a.gif. However, we know that the hidden information in this case is after the end of the a.gif file.

Stego, shown in Figure 7.1, is an interactive graphical interface that allows one to hide text or images into other images, retrieve the secret text or image, and display it, all in the same window. Stego uses the Least Significant Bit insertion technique. This means that every bit of the secret is

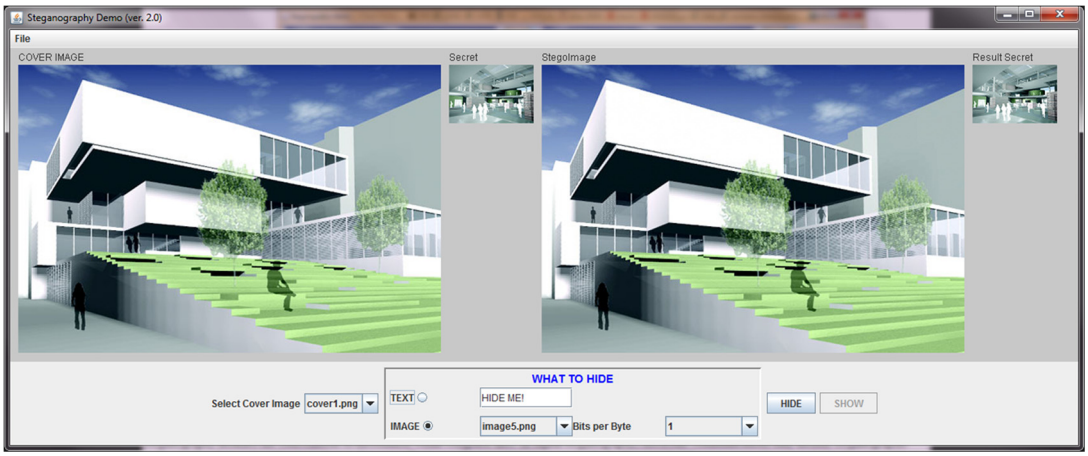

FIGURE 7.1

Graphical interface of the Stego steganography demonstration. application. Four images at the top, from left to right: the cover image, the secret image, the Stego image (output image containing the secret image), and the resulting secret image retrieved from the Stego image. At the *bottom*, the user first selects a cover image, then selects what to hide (text or image) and either types in the text or selects a secret image. The user then selects the number of least significant bits to be used to hide the secret image. Finally, the user can click on the "HIDE" button to create the Stego image and "SHOW" to retrieve the secret image hidden in the Stego image. The user can use the menu bar to save to the disk the intermediate steps (Stego, image, and result image). The user can then compare the images bit by bit with the original images.

inserted in the least significant bit of a byte that represents either the red, green, or blue color of the RGB pixel of the original image (the cover image). Stego also allows the students to hide more than one bit per byte. The effect can instantly be seen on the graphical interface. A user can also save each image on the system to inspect the file size, run a hash function on it, etc., or even load saved Stego images. This feature is provided from the pull-down menu. Figure 7.2 shows how the Stego image looks when the five least significant bits per byte are used to hide the secret image.

For an extra credit problem, we give the students the source code of Stego with one of the functions removed. The function is used to retrieve a secret image from the Stego image. This is the function that the students need to implement for extra credit.

Assignment in a key exchange algorithm

One of the questions students have with symmetric encryption algorithms concerns key distribution. One of the simplest algorithms that we illustrate and describe with real numbers is called Diffie-Helman. The students write a program to show how a "key" can be distributed to two entities without actually transmitting the key itself. The students have the option to work on an extra credit component. The extra credit problem is to use the Diffie-Helman key exchange algorithm to exchange keys in a secure client/server "chatting" application. They exchange a key, then use a symmetric algorithm to establish a secure connection and transmit text and images over the Internet. Later in the semester, after we cover asymmetric encryption and begin talking about Pretty Good Privacy (PGP), they can modify their program so that the key exchange is implemented using RSA.

FIGURE 7.2

A cover image (*left*) and a Stego image (*right*), side-by-side where the five least significant bits per byte are used to hide a secret image. Visible alterations can be observed at the top of the Stego image.

Assignment in asymmetric encryption

RSA [22] and elliptic curve [23] are two popular public key cryptosystems. Public key systems are the standard choice for sending secure information over an insecure channel or network connection. For example, a customer wants to send their credit card information to a website. There is no way for the customer and the Web vendor to share a secret key, as in symmetric encryption systems, without the risk of a third party overhearing the secret key. Instead, public key systems use two keys: a public key that is shared with everyone and a private key that is kept secret by the receiving party. The public key and the private key share a mathematical link, but it is not possible (or is at least computationally very difficult) to derive either key from the other.

RSA is one of the most commonly used public key cryptosystems on the Internet today. It is based on the premise that factoring very large numbers is difficult. "Large" numbers, in modern practice, are 1024-bit or 2048-bit numbers. That is, they are numbers on the order of 2^{1023} ($\sim 10^{307}$) or 2^{2047} ($\sim 10^{615}$). One of these large numbers is chosen to serve as the modulus for encryption and decryption and then a public key exponent and private key exponent are chosen following certain number theory principles. This is the general procedure for generating the associated public and private keys:

1. Choose two prime numbers: p and q.
2. Compute the modulus n: $n = pq$.
3. Compute Euler's totient function $\varphi(n)$: $\varphi(n) = (p-1)(q-1)$.
4. Choose the public key exponent e, such that $1 < e < \varphi(n)$ and e is co-prime with $\varphi(n)$.
5. Compute the private key exponent d, where $(de) \bmod \varphi(n) = 1$.

Step 4 is easily done by selecting a random prime number less than $\varphi(n)$ and then checking that it is not a factor of $\varphi(n)$. Step 5 is somewhat more computationally intensive, but can be computed with the Extended Euclidean algorithm. When this is completed, the public key is the pair (n, e) and the private key is the pair (n, d). Going back to the credit card example above, the Web vendor would send the customer their public key so that the customer can encrypt the credit card information. The private key is kept secret by the vendor. The customer uses standard encoding and padding schemes to produce the message, m, to be sent from the credit card information. To encrypt the message and produce the ciphertext, c, the customer uses this formula:

$$c = m^e \bmod n \tag{1}$$

The encrypted message c is then sent over the Internet and received by the Web vendor, who decrypts the message with this formula:

$$m = c^d \bmod n \tag{2}$$

There are a few different approaches to an RSA assignment. We implement three main components: generating the keys, encrypting and decrypting data, and sending information over a network. The full assignment has students completing all three phases to send a single encrypted file over the network. They first generate the public and private keys, then write a client/server application that allows a user to select a file and transmit it from the client to the server. The server stores the keys and transmits the public key to the client. The client encrypts the file, sends it, and then the server decrypts it.

The first and last steps can be omitted due to time constraints, if necessary. For example, instead of having students write code to generate the keys, they could compute one key pair by hand and use that directly. Alternatively, they could use existing tools such as OpenSSL [24] to generate the RSA keys. The file transfer could also be optional. Students could simply encrypt the file on the local system with one program and decrypt it with another. This would alleviate the need to have code that actually transmits the file, but loses the key exchange between two systems.

Demonstrations

Every student who takes this class is interested in knowing how a virus is written and how viruses work. We demonstrate to the students how four different keyloggers[1] work. We explain the code, which is written in C, and they can experiment with the programs. We also demonstrate three viruses. One of them attacks the browser, another attacks the operating system; of course, we only simulate the deletion and modification of system files. The third virus is a self-replicating virus that infects a machine when a user opens a file. We also demonstrate and explain how to copy DVDs and how they are protected. The code given is written in Java.

Another demonstration involves sending fake email messages (messages where the sender information is modified). A user can hide his or her identity to send emails to others, pretending being someone that the recipient knows. This way, a virus can be distributed. We also write code and utilize open source software to crack passwords and eventually gain system access. Throughout the course, we have the opportunity to talk about ethical issues. However, the students should learn about these techniques so that they can protect themselves in the real world.

Wireless network security

In our course on cryptography and network security, we provide students with solid security fundamentals and hands-on exercises that are focused on wired networks. As a next step, we plan to incorporate wireless security issues into our course. Since wireless networking and communication systems are already fundamental in typical day-to-day use of the Internet, teaching wireless networking and security issues is a demanding task [25,26]. A recent trend in the use of wireless technologies to improve current law enforcement networks demonstrates that wireless security is emerging as a premier research and development issue [27,28]. As the demand increases in law enforcement, hospitals, industry, and the military, so does the importance of wireless and security education for systems and networking professionals.

[1]Keyloggers are special purpose hidden programs that record every keystroke, and sometimes mouse movements, of a user without their knowledge. They then have the capability of sending those keystrokes to a malicious operator so that the user's passwords, credit card numbers, and other personal information can be compromised.

802.11 Wireless security

Despite the popularity of the topic, courses on wireless and security are not frequently offered at the undergraduate level; this is mainly due to the fact that the topic is relatively new and requires further research. Unlike teaching graduate students, teaching a wireless security course to undergraduate students can be a difficult task because the study requires the students to have strong knowledge in both technical implementation and theoretical concepts. Thus, continuous research has been made to develop effective teaching methods [29−33]. Following this trend, we present an overview of hands-on wireless security materials for undergraduate students, with specific examples of 802.11 wireless security experiments.

The first generation of the 802.11 standard included security protection in the form of Wired Equivalent Privacy (WEP) [34]. WEP was found to be vulnerable to various statistical weaknesses, especially in the encryption algorithm it employed to scramble data passed over the Wireless Local Area Network (WLAN) [35,36]. While attempts were made to correct the problem, it is still a relatively simple procedure to crack the protocol. Essentially, one can pull the password right out of the air. In response, the Wi-Fi Alliance stepped up to the challenge and created an interim standard called Wi-Fi Protected Access (WPA). However, WPA's pre-shared key (PSK) mode is also crackable due to a flaw that exists in the authentication procedure [37]. For the most part, a system that involves a password and a user is flawed. The fact that most users select poor passwords provides an opportunity that can be exploited.

802.11 WEP Key cracking experiment

The purpose of this experiment is for the students to learn how to exploit 802.11 wireless security properties. In this project, students use a variety of devices and tools that are widely used in the real market, including Linksys WRT54G series access points (802.11 g broadband routers), OpenWRT [38] firmware, Wireshark [39], and Aircrack-ng [40]. The overall goal is to crack the key of the WEP protocol defined in the 802.11 standard.

To begin this experiment, students have to figure out detailed information about the target wireless network, which includes the MAC address, clients associated with the target access point (AP), security features, etc.

The next step requires students to capture the data traffic on the target network. At this stage, students should have enough information (i.e., weak IVs [41]) to crack the WEP key and be able to associate and utilize the wireless network. Tools such as Airodump-ng [42] can be used. In this experiment, students need to collect enough WEP-encrypted data. For a 64-bit WEP key, between 20,000 and 50,000 packets are required [41,37]. While collecting packets, the WEP key cracking program (aircrack-ng) can be run at the same time. Figure 7.3 shows the cracking output of aircrack-ng.

However, this is a passive approach and requires large amounts of data traffic and data collection time. A more efficient technique is to inject Address Resolution Protocol (ARP) request packets through fake authentication and association [43]. The only known effective way to crack WPA-PSK is to force a re-authentication of a valid client. By forcing the connected client to disconnect, we capture the re-connect and authentication packets (i.e., the four-way handshake), as shown in Figure 7.4.

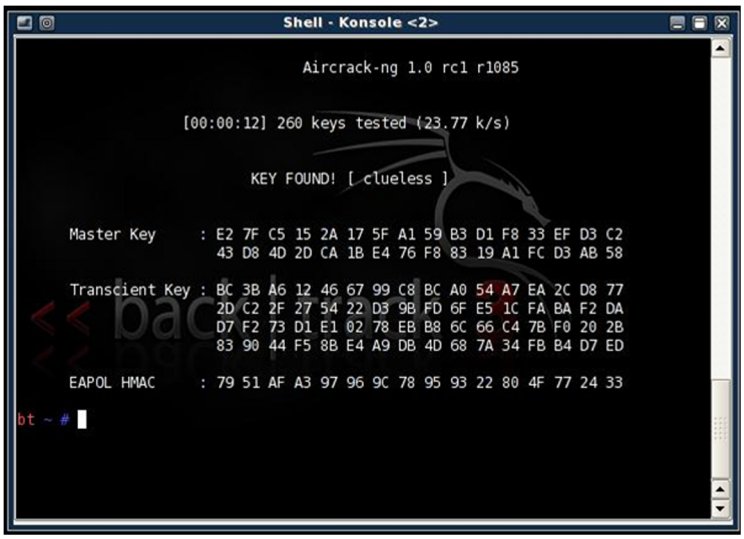

FIGURE 7.3

Cracking output of Aircrack-ng.

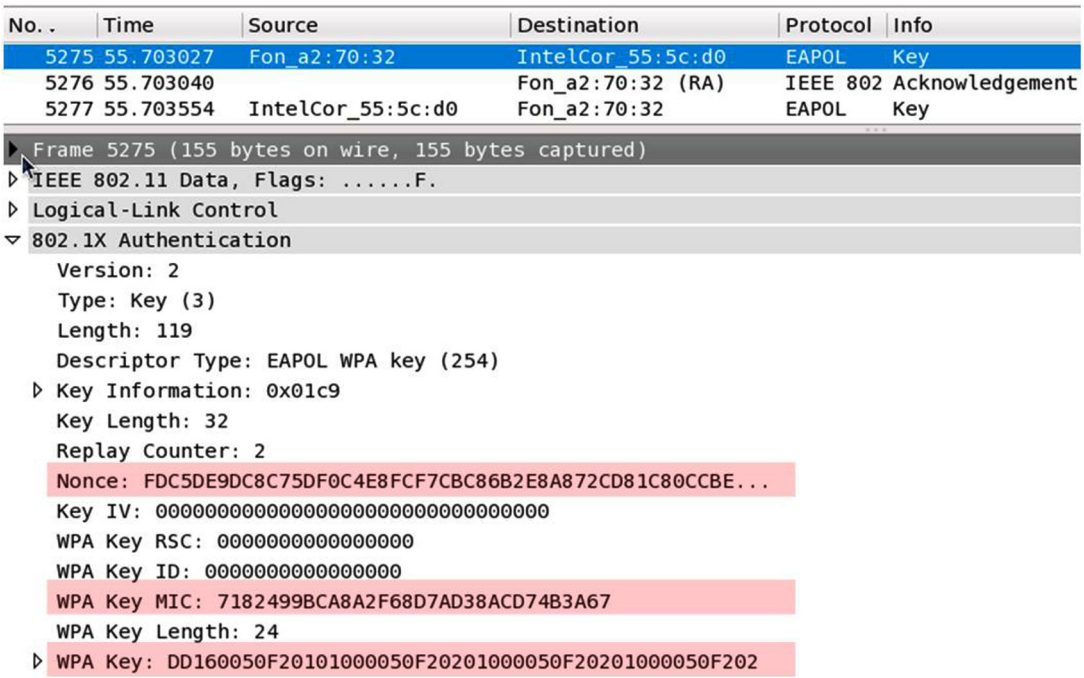

FIGURE 7.4

Wireshark screenshot of a WPA authentication frame.

CONCLUSION

The projects and demonstrations presented in this chapter are excellent activities for students, computing professionals, and law enforcement personnel to begin building practical security experience. In particular, projects are discussed in detail from two major areas of security: cryptography and wireless security.

Cryptographic methods underlie protocols and mechanisms in order to provide secure data transmission and storage. Projects from the three major areas of encryption (hashing, symmetric key, and asymmetric key) are described, showing how they allow students to learn how they work, when it is appropriate to use them, and how they might be vulnerable.

Wireless security is addressed separately, as transmitting signals over the air opens up several different security vectors to consider. The most important one is that wireless signals are broadcast in such a way that any receiver in range can hear and interpret the signal. It is vital that such signals be encrypted and protected at all times. Older techniques such as WEP are susceptible to statistical-based exploits, as demonstrated to students in one project, whereas newer protocols like WPA are more robust.

Together, these two areas cover a broad base of security principles that any computing IT or law enforcement professional should be familiar with, regardless of particular specialization. The projects described are ways to gain that knowledge first-hand by implementing, testing, and sometimes breaking the relevant protocols and services.

References

[1] Yates JK, Voss M, Tsai K. Creating awareness about engineering careers: innovative recruitment and retention initiatives. 29th ASEE/IEEE Frontiers in Education Conference. 1999 Nov;3. San Juan Puerto Rico.

[2] Tester JT, Scott D, Hatfield J, Decker R, Swimmer F. Developing recruitment and retention strategies through "Design4Practice" curriculum enhancements. 34th ASEE/IEEE Frontiers in Education Conference; 2004 Oct; Savannah, GA, USA.

[3] Tester JT, Hatfield J. The Design4Practice sophomore design course: Adapting to a changing academic environment. ASEE Annual Conference Proceedings; 2005 June.

[4] Anderson-Rowland MR, Blaisdell S, Fletcher S, Fussel P, Jordan C, McCarthey MA, et al. Comprehensive programmatic approach to recruitment and retention in the college of engineering and applied sciences. In: Ptoc of the 29th Annual Frontiers in Education Conference. 1999;1.

[5] Courter SS, Millar SB, Syons L. From the students' point of view: experiences in a freshman engineering design course. Journal of Engineering Education 1998;87(3):283–7.

[6] Wang H. From C to Python: Impact on retention and student performance. In: Proc of The 2008 International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS'09); 2009 Jul 13-16; Las Vegas NV, USA. p.170–4.

[7] Tanaka JC, Gladney LD. Strategies for recruiting and retaining minorities in physics and biophysics. Biophys J 1993;65:552–8.

[8] Hatfield JM, Tester JT. LEGO plus. Proceedings of the 2005 American Society for Engineering Education Annual Conference & Exposition; 2005.

[9] Parcover JA, McCuen RH. Discovery approach to teaching engineering design. Journal of Professional Issues in Engineering Education and Practice 1995:236–41.
[10] Seymour E, Hewitt NM. Talking about leaving: Why undergraduates leave the sciences. Boulder, CO: Westview Press; 1997.
[11] Twigg CA. Using asynchronous learning in redesign: Reaching and retaining the at-risk student. JALN 2004;8(1):7–15.
[12] Glumich SM, Kropa BA. DefEX: Hands-on cyber defense exercises for undergraduate students. In: Proc of the 2011 Int Conf on Security and Management (SAM'11); 2011 July; USA.
[13] US Department of Commerce/National Institute of Standards and Technology, FIPS PUB 197; Advanced encryption standard (AES). Federal Information Processing Standards Publication, 2001. [Internet]. Available at: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.
[14] Rivest RL. The MD4 message-digest algorithm. Crypto, LNCS 1991;537:303–11.
[15] http://www.codeproject.com/Articles/11643/Exploiting-MD5-collisions-in-C.
[16] MD5 Collision Demo [homepage on the Internet]. [Updated Oct. 11 2011; cited 2013 Sep. 11]. Available from: <http://www.mscs.dal.ca/selinger/md5collision/>.
[17] Kipper G. Investigator's guide to steganography. Print ISBN: 978-0-8493-2433-8 eBook ISBN: 978-0-203-50476-5. Auerbach Publications; 2004.
[18] Ptzmann B. Information hiding terminology. First Workshop of Information Hiding Proceedings. Lecture Notes in Computer Science. 1996 May 30-Jun 1; Cambridge, UK. Springer-Verlag. 1996;1174:347–50.
[19] Choche A, Arabnia HR. A methodology to conceal QR codes for security applications. Proceedings of the International Conference on Information and Knowledge Engineering (IKE'11); 2011 Jul; USA.
[20] Hide At End Demo Software [homepage on the Internet]. [Cited 2013 Sep. 11]. Available from: <http://faculty.cs.wit.edu/Bldeligia/PROJECTS/HAE/index.html>.
[21] Stego Demo Software [homepage on the Internet]. [Cited 2013 Sep. 11]. Available from: <http://faculty.cs.wit.edu/Bldeligia/PROJECTS/Stego/index.html>.
[22] Rivest R, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 1978;21:2.
[23] Black I, Seroussi G, Smart N. Elliptic curves in cryptography. Cambridge University Press; 1999.
[24] OpenSSL Project [homepage on the Internet]. [Cited 2013 Sep. 11]. Available from: http://www.openssl.org/.
[25] Ma D, Tsudik G. Security and privacy in emerging wireless networks. [Invited Paper]. Wireless Commun IEEE 2010;17(5):12–21.
[26] Shiu YS, Chang SY, Wu HC, Huang SCH, Chen HH. Physical layer security in wireless networks: a tutorial. Wireless Commun. IEEE 2011;18(2):66–74.
[27] Thomas P, Cloherty J, Ryan J. ABC News. After $350 million, law enforcement wireless network success still doubtful. [Internet]. 2012 Jan. Available at http://abcnews.go.com/.
[28] Seth M, Kasera SK, Ricci RP. Emergency service in Wi-Fi networks without access point association. In: Proceedings of the first International Conference on Wireless Technologies for Humanitarian Relief (ACWR '11). ACM, New York, NY; 411–19.
[29] Sarkar NI, Craig TM. Teaching wireless communication and networking fundamentals using Wi-Fi projects. IEEE Transactions on Education. 2006;49(1):98–104.
[30] Güzelgöz S, Arslan H. A wireless communications systems laboratory course. IEEE Trans Educ 2010;53(4):532–41.
[31] Lin Y. An ultra low cost wireless communications laboratory for education and research. IEEE Transactions on Education 2012;55(2):169–79.
[32] Chenard JS, Zilic Z, Prokic M. A laboratory setup and teaching methodology for wireless and mobile embedded systems. IEEE Transactions on Education 2008;51(3):378–84.

[33] Mateo Sanguino TJ, Serrano Lopez C, Marquez Hernandez FA. WiFiSiM: an educational tool for the study and design of wireless networks. IEEE Trans Educ 2013;56(2):149–55.
[34] IEEE standard for information technology - telecommunications and information exchange between systems- local and metropolitan area networks - specific requirements -Part II: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. ANSI/IEEE Std 802.11. 2003.
[35] Boland H, Mousavi H. Security issues of the IEEE 802.11b wireless LAN. Canadian conference on electrical and computer engineering; 2004. 333–36.
[36] Fluhrer SR, Mantin I, Shamir A. Weaknesses in the key scheduling algorithm of RC4. In: Vaudenay S, Youssef AM, editors. Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography (SAC '01). London, UK: Springer-Verlag; p. 1–24.
[37] Berghel H, Uecker J. WiFi attack vectors. Commun ACM 2005;48(8):21–8.
[38] OpenWRT [homepage on the Internet]. [Cited 2013 Sep. 11]. Available from: https://openwrt.org/.
[39] Wireshark, [homepage on the Internet]. [Cited 2013 Sep. 11]. Available from: http://www.wireshark.org/.
[40] Aircrack-ng key craking program, [homepage on the Internet]. [Cited 2013 Sep. 11]. Available from: http://www.aircrack-ng.org/.
[41] Reddy SV, Sai Ramani K, Rijutha K, Ali SM, Reddy CP. Wireless hacking - a WiFi hack by cracking WEP. 2010 Second international conference on education technology and computer (ICETC); 2010. 189–93.
[42] Airodump-ng tool, [homepage on the Internet]. [Updated: 2013 Aug. 16; Cited 2013 Sep. 11]. Available from: http://www.aircrackng.org/dok u.php?id5airodump-ng.
[43] Yuan X, Wright OT, Yu H, Williams KA. Laboratory design for wireless network attacks. In: Proceedings of the fifth annual conference on information security curriculum development (InfoSecCD '08). ACM, New York, NY; 5–12.

Further Reading

Deligiannidis L. Classroom experiences: Disallowing laptops during lectures improves student learning. In Proc of The 2011 International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS'11); 2011 Jul 18–21; Las Vegas, NV. p. 217–221.