# Network Security Course: A Demonstration of Project-Based Learning

Leonidas Deligiannidis, Charlie Wiseman, Mira Yun, Tom Goulding

Wentworth Institute of Technology
Department of Computer Science and Networking
550 Huntington Avenue
Boston, MA 02115, USA
{deligiannidisl, wisemanc, yunm, gouldingt}@wit.edu

## ABSTRACT

Systems and Network Security is a very important topic in today's world where Information Technology is becoming part of our everyday lives. We access and interact with numerous services on the Internet that require us to exchange sensitive data. The Internet is a highly distributed system operated by many different entities and as such should not be trusted by end users. This is the reason that we need strong security measures to communicate in a secure way, privately, with other (trusted) entities. Users, whether consumers or businesses, retain no control how their information is routed among the many networks that comprise the Internet. Therefore, there is a strong need for cryptographic protocols to authenticate, gain trust, and establish a secure channel for exchanging data.

We are enriching our curriculum by offering several Computer Science advanced electives each semester (Fall, Spring and Summer). Students have the option of taking advanced elective courses that fit their interest. One of our newer courses, Cryptography and Network Security, addresses these issues of communicating securely over inherently insecure channels such as the Internet. In this paper we present the material taught in this class as well as the assignments and demonstrations utilized in the class room to motivate the students in learning.

**Keywords -** Network Security, Education

## 1. INTRODUCTION

Wentworth Institute of Technology is a baccalaureate degree granting institution. The Department of Computer Science and Networking has been in existence since the early nineteen eighties. Our department offers two degrees: one leading to a Bachelor of Science in Computer Science and another leading to a Bachelor of Science 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 wide skill-set in networking, computer science, and management. Both majors select four advanced Computer Science electives from a pool of 10-12 courses during their Junior and Senior years. One of the courses now offered regularly is the Cryptography and Network Security course. This course has been very successful, with excellent student engagement and feedback [21]. The material presented is difficult, but demonstrations using numerical and/or graphical examples engage the students. In fact, many of the students work on the more difficult extra credit assignments in addition to the regular course assignments. Many students ultimately make a carrier change because of this course.

The book used in the class is "Cryptography and Network Security: Principles and Practice", by William Stallings, Prentice Hall. This is an excellent book and has been adopted by many professors offering similar courses at the college level. Most programming assignments are done on students' laptop computers using the Java language; a few of the assignments are done in C#. There are six programming assignments where we use symmetric encryption, digital signatures, and digital certificates, and two more assignments where we implement RSA public key cryptography from the ground up. There are two exams during the semester and a final exam at the end of the semester. Because of the popularity of the topics covered in class, we offer this course during the Summer semester every year and sometimes during the Spring as well.

In the rest of the paper we will focus on the programming assignments, demonstrations, and future material we plan on incorporating into this course.

## 2. BACKGROUND

It is suggested that hands-on, investigative teaching with associated exercises improves the learning performance of the students [17]. It is also shown that engaging students in hands-on exercises promotes active learning and helps students develop critical thinking skills [19][20] more 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 [3][7][11][16][18][19][20][23]. A three year study was performed to determine why STEM majors switch to non-STEM majors [27]. They found that students switch majors because of lack of interest, teaching methodology ineffectiveness, and because they felt 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 [6].

DefEx is a set of hands-on cyber-defense exercises that focuses on undergraduate development through understanding and problem solving related to security [28]. 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.

## 3. COURSE WORK

In this 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.

### *3.1. Assignment in Symmetric Encryption*

The Data Encryption Standard (DES) algorithm was 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 to be insecure and has already been superseded by the Advanced Encryption Standard (AES) [30].

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 file. In another assignment the students compare the performance of DES and AES by plotting how fast different size files are encrypted and decrypted.

### *3.2. 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 if you only have 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 cryptographic hash function *H*: *H(M) != H(M')*. Many cryptographic hash functions are based on the so called MD4 algorithm initially proposed in [24], 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. Then, we demonstrate how they can break MD5 using the techniques described in [12][13]. 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.

### *3.3. Extra Credit Assignment on Steganography*

Steganography is a technique to conceal information so that only the communicating parties know about the existence of the information in any form [15]**.** Steganography uses a Cover Image which is an image that will have information embedded in it. Then there is the actual information itself that 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 [4] is obtained. It is also possible to encrypt the information before embedding it, thereby adding another level of security [2].

We demonstrate two applications that use steganography. Both of them are written in Java and are accessible via webstart technology: HAE (Hide at End) [9] and Stego [29]. 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.

Stego, shown in Figure 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 them all in the same window. Stego uses the Least Significant Bit insertion technique. This means that every bit of the secret is 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 be seen instantly on the graphical interface. A user can also save each image on the system to inspect the file size, run a hash function on them, etc, or even load saved Stego Images. This feature is provided from the pull down menu. Figure 2 shows how the Stego Image looks when the 5 least significant bits per byte are used to hide the secret image.

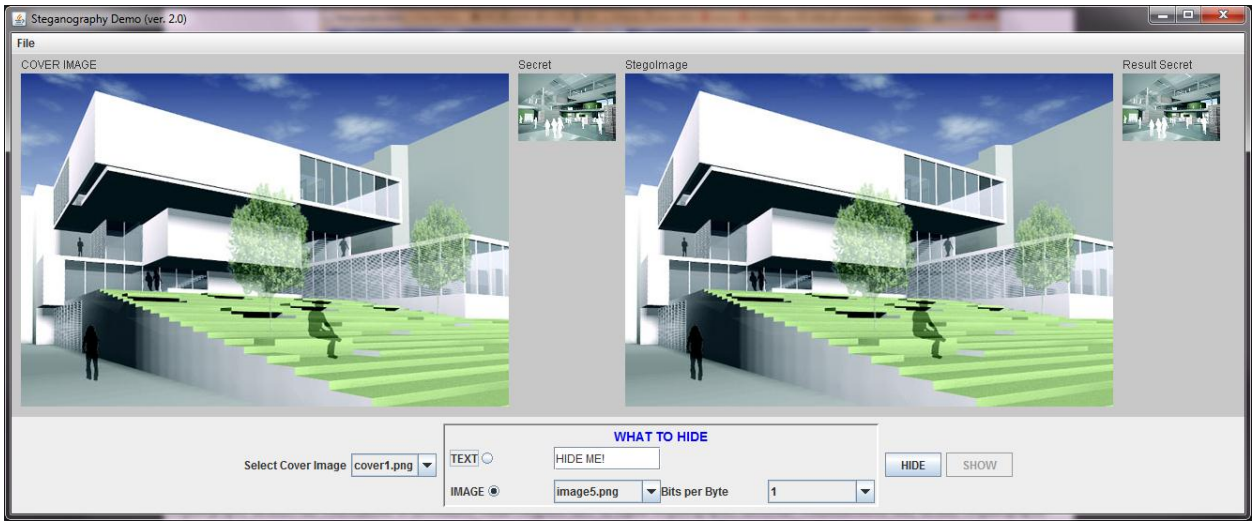


**Figure 1.** Graphical interface of the "Stego" steganography demonstration application. Four images at the top, from left to right: the Cover Image, the secret image, the Stego Image (output image containing the secret image), and the result secret image retrieved from the Stego Image. At the bottom, the user first selects a cover image, then the user selects what to hide (text or image) and either types in the text or selects a secret image. Then, the user 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.

**Figure 2.** A Cover Image (left) and a Stego Image (right), side-by-side where the 5 least significant bits per byte are used to hide a secret image. Visible alterations can be observed at the top of the Stego Image.

For extra credit, we give the source code of Stego to the students with one of the functions is removed. The function is used to retrieve a secret image from the Stego Image and must be replaced by the students.

### *3.4. Assignment in a Key Exchange Algorithm*

One of the questions students have with symmetric encryption algorithms is key distribution. One of the simplest algorithms that we illustrate and describe with real numbers is 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 is to use the Diffie-Helman key exchange algorithm to exchange keys in a secure client/server "chatting" application. They exchange a key and then they 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 we begin talking about PGP, they can modify their program so that the key exchange is implemented using RSA.

### *3.5. Assignment in Asymmetric Encryption*

RSA [25] and elliptic curve [14] 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 web site. There is no way for the costumer 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 computational very difficult) to derive either key from the other.

RSA is one of the most commonly used public key cryptosystems in 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 done easily 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:

6) $c = m^e \bmod n$

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

7) $m = c^d \bmod n$

There are a few different approaches to an RSA assignment. There are 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, and 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 [22] 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.

### *3.6. Demonstrations*

Every student that takes this class is interested in knowing how a virus is written and how they work. We demonstrate to the students how four different Keyloggers work. We explain the code, which is written in C, and they can experiment with the programs. We also demonstrate two viruses. One of them attacks the browser and the other attacks the operating system (of course we only simulate the deletion and modification of system files). 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). 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.

## 4. CONCLUSIONS AND FUTURE WORK

We provide students with solid security fundamentals and hands-on exercises that are focused on wired networks through our course on Cryptography and Network Security. As a next step, we will incorporate wireless security issues into our course.

Wireless enabled devices such as laptops, smartphones, and tablets are becoming increasingly affordable and accessible. Wireless technology has become a crucial component in computer networking and security technology [8][32]. 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 it is still a relatively new topic that requires greater research. In our course, we will focus on security issues relating to current Wi-Fi (802.11) systems.

The first generation of the 802.11 standard included security protection in the form of Wired Equivalent Privacy (WEP) [1]. WEP was found to be vulnerable to various statistical weaknesses, especially in the encryption algorithm it employed to scramble data passed over the WLAN [5][26]. 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 [10]. For the most part, if there is a password and a user is involved then the system is flawed. This fact, combined with the reality that most users select poor passwords, provides an opportunity that can be exploited.

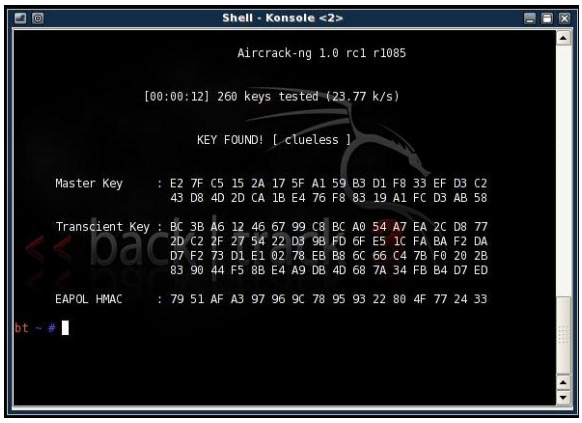


**Figure 3.** Cracking output of Aircrack-ng.

In order to provide hands-on learning experiences in wireless security, we will use devices and tools that are widely used in the real market. Linksys WRT54G series access points (802.11g broadband routers) and OpenWRT [31] firmware will be used for deploying and managing Wi-Fi networks. As shown in Figures 3 and 4, we will explore the vulnerabilities of the WEP and WPA-PSK protocols with key cracking tools and network protocol analyzers.

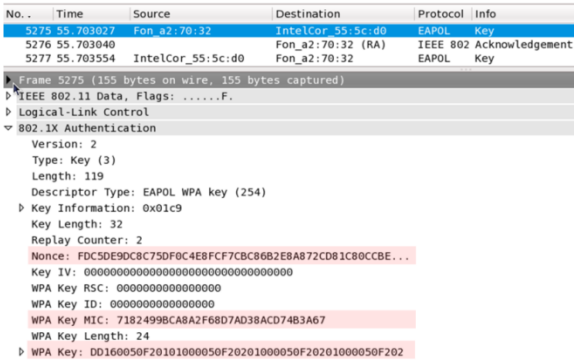


**Figure 4.** Wireshark screenshot of a WPA authentication frame.

Students will learn two WEP key cracking schemes: passive traffic collection and active injection. As a very basic approach, minimal WEP encrypted data will be passively collected. Approximately 20,000 to 50,000 packets are required to crack a 64 bit WEP key. From the collected data packets, we will use weak Initialization Vectors (IVs) to crack the WEP key as defined in the 802.11 standard. However this passive approach requires large amounts of data traffic and collecting time. As a more efficient technique, we can inject ARP request packets through fake authentication

and association. 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 4.

Although students are mainly excited to learn about hacking and cracking skills, we believe that this course will allow students to be more engaged in the topic of wireless security. It will become an even better starting point for students to develop their interest in the topic of network security.

## REFERENCES


[1] "IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE Std 802.11-1997 , vol., no., pp.i-445, 1997.

[2] Akshay Choche and Hamid R. Arabnia, "A Methodology to Conceal QR Codes for Security Applications". Proceedings of the International Conference on Information and Knowledge Engineering (IKE'11): July 2011, USA.

[3] Anderson-Rowland, M. R.; Blaisdell, S., Fletcher, S., Fussel, P., Jordan, C., McCarthey, M. A., Reyes, M. A., White, M., "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, Vol.(1), 1999.

[4] Birgit Ptzmann, Information Hiding Terminology, First Workshop of Information Hiding Proceedings, Cambridge, U.K. May 30 - June 1, 1996. Lecture Notes in Computer Science, Vol.1174, pp 347-350. Springer-Verlag (1996).

[5] Boland, H.; Mousavi, H.; "Security issues of the IEEE 802.11b wireless LAN," Electrical and Computer Engineering, 2004. Canadian Conference on, vol.1, no., pp. 333- 336 Vol.1, 2-5 May 2004.

[6] Carol A. Twigg, "Using Asynchronous Learning in Redesign: Reaching and Retaining the at Risk Student", JALN Vol.(8), Issue 1, Feb. 2004 pp 7-15.

[7] Courter, S. S.; Millar, S. B.; Syons, L. "From the students' point of view: Experiences in a freshman engineering design course" Journal of engineering education, Vol.(87), no(3), pp 283-287, Jul 1998.

[8] D Ma; Tsudik, G.; , "Security and privacy in emerging wireless networks [Invited Paper]," Wireless Communications, IEEE , vol.17, no.5, pp.12-21, October 2010.

[9] HAE website http://faculty.cs.wit.edu/~ldeligia/PROJECTS/HAE/index.html

[10] Hal Berghel and Jacob Uecker, "WiFi attack vectors". Commun. ACM 48, 8 (August 2005)

[11] Hong Wang, "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), pp.170-174, July 13-16 2009 Las Vegas NV, USA.

[12] http://www.codeproject.com/Articles/11643/Exploiting-MD5-collisions-in-C

[13] http://www.mscs.dal.ca/~selinger/md5collision/

[14] I. Black, G. Seroussi and N. Smart. "Elliptic Curves in Cryptography", Cambridge University Press, 1999.

[15] Investigator's Guide to Steganography Gregory Kipper Auerbach Publications 2004. Print ISBN: 978-0-8493-2433-8 eBook ISBN: 978-0-203-50476-5.

[16] Jacqueline C. Tanaka, Larry D. Gladney, "Strategies for Recruiting and Retaining Minorities in Physics and Biophysics", Biophysical Journal Vol.(65), Jul. 1993 pp552-558

[17] Janet K. Yates, Madeleine Voss, Kuei-wu Tsai, "Creating Awareness about Engineering Careers: Innovative Recruitment and Retention Initiatives" 29th ASEE/IEEE Frontiers in Education Conference, Vol. 3, Nov 1999, San Juan Puerto Rico.

[18] Jerry M. Hatfield, John T. Tester, "LEGO Plus", Proceedings of the 2005 American Society for Engineering Education Annual Conference & Exposition, 2005.

[19] John T. Tester, David Scott, Jerry Hatfield, Rand Decker, Fonda Swimmer, "Developing Recruitment and Retention Strategies through "Design4Practice" Curriculum Enhancements", 34th ASEE/IEEE Frontiers in Education Conference, Oct. 2004, Savannah, GA, USA.

[20] John T. Tester, Jerry Hatfield; "The Design4Practice Sophomore Design Course: Adapting to a Changing Academic Environment," ASEE Annual Conference Proceedings, June 2005.

[21] Leonidas Deligiannidis, "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), pp.217-221, July 18-21 2011, Las Vegas NV, USA.

[22] OpenSSL website. http://www.openssl.org/

[23] Parcover, J. A., McCuen, R. H., "Discovery Approach to Teaching Engineering Design",

Journal of Professional Issues in Engineering Education and Practice, pp 236-241, Oct. 1995.
[24] R. L. Rivest, The MD4 message-digest algorithm, Crypto, LNCS 537 (1991) 303-311.
[25] R. Rivest, A. Shamir and L. Adleman, ”A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”, Communications of the ACM, 21(2), 1978.
[26] Scott R. Fluhrer, Itsik Mantin, and Adi Shamir, “Weaknesses in the Key Scheduling Algorithm of RC4”, In Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography (SAC '01), Serge Vaudenay and Amr M. Youssef (Eds.). Springer-Verlag, London, UK, 1-24.
[27] Seymour, E. & Hewitt, N.M., “Talking about leaving: Why undergraduates leave the sciences”. Boulder, Co: Westview Press 1997.
[28] Sonja M. Glumich and Brian A. Kropa," DefEX: Hands-On Cyber Defense Exercises for Undergraduate Students" In Proc. of the 2011 Int. Conf. on Security and Management (SAM'11), July 2011, USA.
[29] Stego website http://faculty.cs.wit.edu/~ldeligia/PROJECTS/Stego/index.html
[30] U.S. Department of Commerce/National Institute of Standards and Technology, FIPS PUB 197, Advanced Encryption Standard (AES), Federal Information Processing Standards Publication, 2001. Available at http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.
[31] Website of OpenWRT https://openwrt.org/
[32] Yi-Sheng Shiu; Shih Yu Chang; Hsiao-Chun Wu; Huang, S.C.-H.; Hsiao-Hwa Chen; , "Physical layer security in wireless networks: a tutorial," Wireless Communications, IEEE , vol.18, no.2, pp.66-74, April 2011.