

# An Innovative Modular Approach of Teaching Cyber Security across Computing Curricula

Akhtar Lodgher, Jeong Yang, Ummugul Bulut\*

Department of Computing and Cyber Security

\*Science and Mathematics

Texas A&M University – San Antonio

San Antonio, Texas, USA

{ALodgher, JYang, UBulut}@tamusa.edu

**Abstract:** This work in progress paper in the innovative practice category presents an innovative approach where cybersecurity concepts and methods are taught as individual modules across several existing courses in the curriculum of computer science (CS), computer information systems (CIS) and information technology (IT) courses. In almost all universities, cyber security is currently taught as an “add-on” track or concentration where students take a series of three to five courses related to cyber security in their junior and senior years after they have completed a series of required core courses [4]. Cyber security is so important that it can no longer be just a topic or a track – it is the way in which ALL software should be written. It should be taught in almost every course in a computer science curriculum because cyber security affects every major software component in any computing system. The modules are designed to be independent and loosely coupled, which enables them to be integrated into courses offering similar course content within the program, other computing related programs, or courses offered by other colleges and universities. The modules introduce students to the various concepts of cyber security from the freshman year, to ensure that fundamental programming concepts are taught from a security perspective – building a strong cyber security foundation for future courses. This innovative approach of teaching cyber security across various computing curricula will widen the pipeline to meet the demand of the nation for cyber security educated professionals.

**Keywords**—cyber security education, modular approach, cyber security foundation, Caesar cipher

## I. INTRODUCTION

The goal of this paper is to present the work in progress of developing modules for introducing and teaching cyber security concepts in all computing curricula. Teaching cyber security is so important that it no longer can be taught as an “add-on” track or concentration. It must be taught in all the major core, elective and minor courses of the various computing (CS, CIS, AS and even high school) curriculum. The main objectives of the entire project – funded through a grant from National Security Agency (NSA) [1] are:

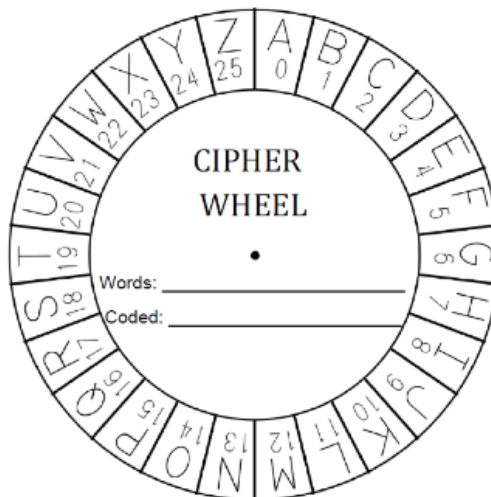
1. Develop curricula and teaching material modules for eight computer science courses, beginning with the first course in computing, and leading to the senior level courses [3].
2. Develop a corresponding series of hands-on code modules beginning with simple concepts and leading to the depth of advanced concepts, to ensure that students get a deeper

understanding of the concept by actually implementing the concept through code, testing it, and answering the exercise questions.

3. Incorporate the NICE NCWF [2] topics of (i) Cyber threat and vulnerabilities, (ii) Risk Management, (iii) Software reverse engineering, and (iv) Cryptography into the curricula of these courses.
4. Keep the code modules compact, complete and independent, to ensure that they can be used in other courses where the topic may be covered as just a single topic, such as an elective course.

Forty one modules on the various topics of cyber security are being developed for the funded project. This paper reports the work in progress of eight of those modules related to introductory cryptography and network communications, and the preliminary results of developing and using these modules.

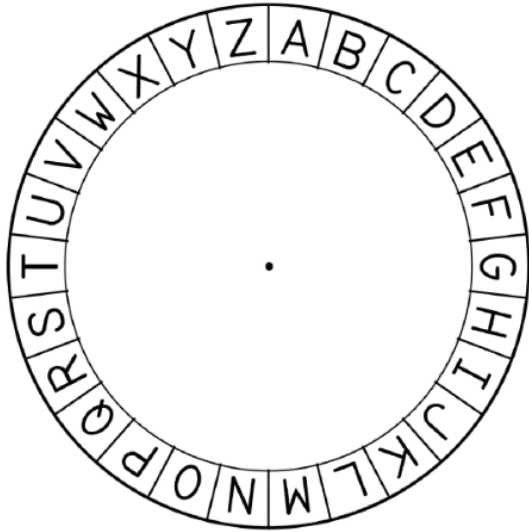
## II. MODULE FOR TEACHING ENCRYPTION IN THE FIRST COURSE



**Figure 1:** Caesar Cipher inner wheel for encrypted text

This module shows how Caesar Cipher is used for encoding simple ASCII text. The concept of Caesar cipher is explained using hands-on exercises and the code for implementing Caesar Cipher is demonstrated. A lab exercise is included to ensure that the concept of Caesar cipher is understood.

### A. Encryption description and Student Lab Activity



**Figure 2:** Caesar Cipher outer wheel for original text

The module begins with a general explanation of encryption, shift encryption, and Caesar Cipher followed by a student activity where students must print the cipher wheels provided in the printable pdf file (*Figures 1 and 2*). The wheels are cut using a pair of scissors and laid on each other with the larger wheel at the bottom. The center of the wheels are pierced, and held together with a paper clip. If time is limited, students may be given prepared wheels.

Students are given instructions for encrypting and decrypting text using these wheels. After the students have done a few exercises and are comfortable using the wheel, a few assessment questions are presented such as: the importance of encryption, shift based encryption, etc.

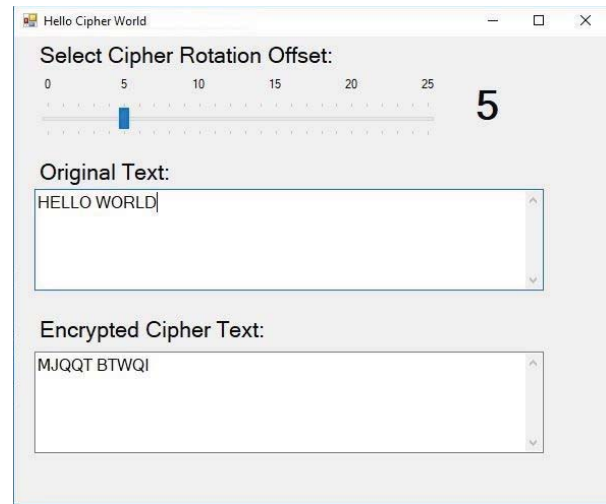
### B. Software lab activity for simple encryption

The algorithm is briefly explained using a PowerPoint presentation. The level of the explanation is determined based on the concepts already taught. If it is the first class ("Hello World level"), the loop structure is just mentioned, but for slightly advanced students, the loop structure is explained in detail. The students are shown a computational working version of the code as shown in *Figure 3*.

For students who already know the loop structure, the code behind the encryption is explained. The students are then given a blank template of the above code and they have to fill the code segment to get the software working completely. A few assessment question such as decryption of encrypted text, extensions to include numbers, etc are asked to gauge the level of understanding of the code.

## III. MODULES FOR DIVISIBILITY OF INTEGERS, THEIR PROPERTIES, AND EUCLIDEAN ALGORITHM FOR GCF

The modules described in this section are for classes in which the basics of cryptography are introduced.



**Figure 3:** Caesar Cipher interactive interface for encryption and decryption

### A. Divisibility of Integers and their Properties

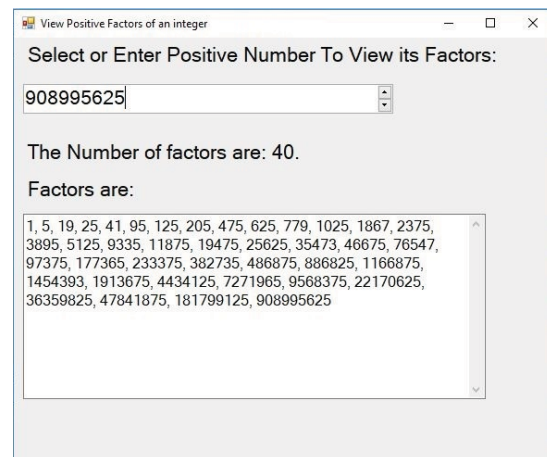
The purpose of these modules is to teach the divisibility of integers. More specifically, the objectives are for students to: (a) learn divisibility and specific terminology used along with the divisibility algorithm. (b) be able to generate the code to find the factors of a given number (c) learn important divisibility properties along with their mathematical proofs (d) analyze the code to better understand divisibility rules.

This lesson includes two important properties of divisibility. If  $a$ ,  $b$  and  $c$  are integers, then:

If  $a|b$  and  $b|c$ , then  $a|c$

If  $a|b$  and  $a|c$ , then  $a|(mb+nc)$  for arbitrary  $m$  and  $n$ .

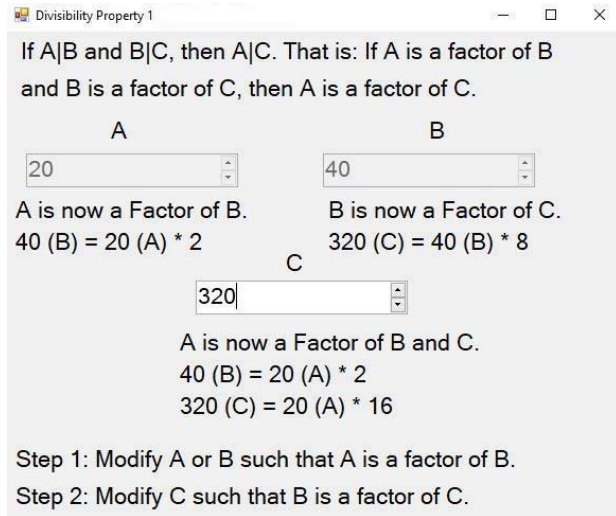
Using PowerPoint slides, the concept of divisibility and factors is explained. An assessment of the understanding of these concepts is done by asking students to find the factors of small integers. While the manual process of finding factors is workable for small integers, the process is very tedious for large integers. The computational algorithm for computing factors is introduced and students are given the executable of the program (*Figure 4*) to view the factors of integers.



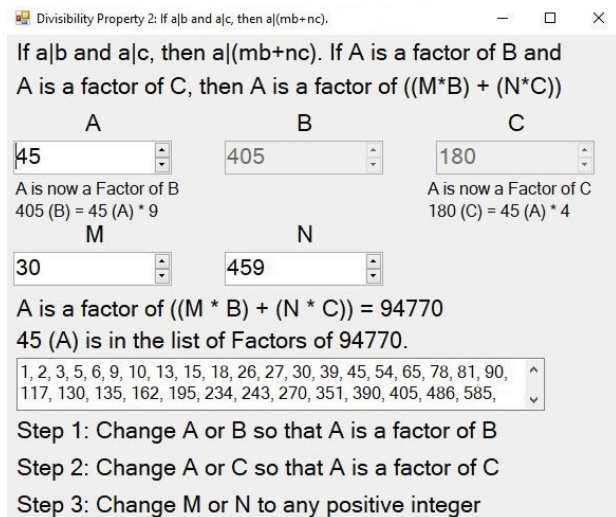
**Figure 4:** Interactive Interface to view factors of integers

The code behind the algorithm is explained and the students are then given a blank template of the above code and they have to fill the code segment to get the software to display the factors.

The divisibility properties of integers are explained and the students work with several examples of using large integers, by using the interactive interfaces shown in *Figures 5 and 6*.



**Figure 5:** Interactive Interface to view divisibility property 1

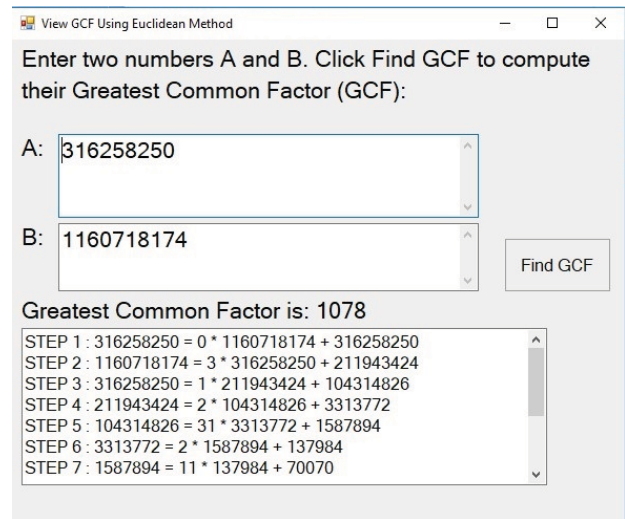


**Figure 6:** Interactive Interface to view divisibility property 2

#### B. Module for computing Greatest Common Factor (GCF)

This module is used to help a student understand and generate the greatest common factor of two numbers using the Euclidean algorithm. Again, students are explained the Euclidean algorithm and several exercises are worked in class for small integers. However, as the size of an integer grows larger, a computational implementation of the algorithm is needed. The interactive interface of this module (*Figure 7*) is used to determine the greatest common factors of large numbers. Once a student understands the working of the computational algorithm, the template code of the above

interface is given to them. They have to complete the code template to get the interface to work properly. Assessment questions are then asked, such as, increasing the efficiency of the implementation, iterative vs recursive implementation, and exception handling when computational limits may be reached.



**Figure 7:** Interactive Interface to view GCF of two integers

#### IV. MODULES FOR DEVELOPING CLIENT SERVER COMMUNICATIONS AND SNIFFING THE COMMUNICATIONS

A series of three modules are built to establish simple client-server communications and to sniff the communications. A simple service (server) is developed (using the server listener) to listen to client requests on user defined ports. The communications messages use the HTTP protocol, for simplicity, to mimic web communications.

##### A. Developing the client/server communications and testing it

This module shows how socket communications is used for communications between a client and a server. The TCP (connection oriented protocol) is used. A simple web server and a simple web client are developed in C# to communicate using the TCP protocol. A lab exercise is included to reinforce the communication using a web server and a web client protocol for communications.

The socket communications on the listening service (server) is set up using the following steps: (a) start the service (b) create a socket (c) wait for the client to connect (d) receive data from client (e) respond to the client (f) check if connection is to be terminated with client (based on timeout or client request) (g) if No, go back to step c and wait for client. Students are given the pseudo code of the above steps and they complete the code for the above process (*Figure 8*).

The socket communications of the client is set up similarly using the following steps: (a) start the application (b) determine the server address and server port number with which to communicate (c) create a string of text (request) to send to server (d) create a socket (TCPClient) specific to the server and the port (e) create a network stream to send the data



(f) convert the request to a stream by putting it into the stream buffer and send it (g) close the stream and the client.

```
1. Enter Server Port number
2. Start Server
3. Exit.

EnterChoice (Number between 1 and 3):1

Port Number :12345
Port Number = 12345
1. Enter Server Port number
2. Start Server
3. Exit.

EnterChoice (Number between 1 and 3):2
Server Started on Port Number: 12345
```

**Figure 8:** Listener setup of server

Again, the students are given the pseudo code of the client setup steps and they complete the code. Once the client and server code are setup, they are tested thoroughly for sending and receiving communications. The text exchanged is then modified to match the HTTP protocol to mimic the web traffic for most common web page transfers (*Figure 9*).

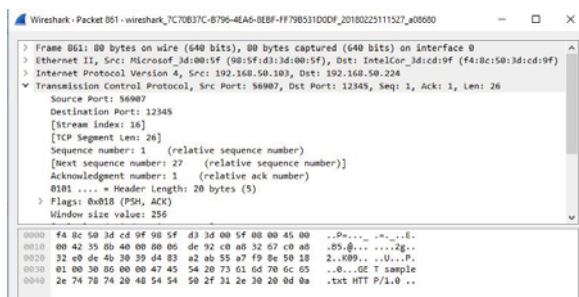
```
1. Enter Server address
2. Enter Server Port number
3. Get a text file and display to screen (using the GET method)
4. Get a text file and save file on the client (using the GET method)
5. Get only the header information of file (using HEAD Method)
6. Send a text file to a server to be saved at the server (use the PUT method)
7. Exit.

EnterChoice (Number between 1 and 7):
```

**Figure 9:** Setup of client using HTTP-like protocol

### B. Sniffing the traffic

The traffic between the client and server is sniffed using tools such as Wireshark. A sample snapshot of sniffing the GET command of the HTTP protocol is shown in *Figure 10*.



**Figure 10:** Intercept of client-server communications

There are several more modules in this section which are under development. Those modules will systematically build the learning, on this basic client server communications, to examine the vulnerabilities in the communications and attacking the communications to exploit them. A common

vulnerability is time and resource allocation by the listener for every client connection, which is exploited by the Denial-of-service (DOS) and the Distributed DOS (DDOS) attacks.

## V. PRELIMINARY RESULTS OF USING THESE MODULES

As part of the grant requirements for assessment of the created modules, a formative evaluation methodology is used. The module materials are reviewed by students, hired on grant funds. The initial topic knowledge level of the student assessing the module is expected to be the same as the knowledge required by the module. The student follows the instructional materials and the Powerpoint slides, and then attempts to complete the exercises on their own. For any difficulties they encounter, they take notes and bring it to the attention of the investigators – for example – clarity of explanation, code environment, etc. The notes are reviewed by the investigators and appropriate corrections are made. A summative evaluation of the modules will be done by researchers of the funding agency once they have received all the modules from all the investigators.

The Caesar cipher encryption module has been used to demonstrate the encryption at two local high schools. At one high school (Brooks) it was demonstrated to students primarily interested in computing careers and they were given the prepared wheel and asked to encrypt their own text. The interactive environment was then used to “break” the code – determine the shift. At another high school (Palo Alto) the Caesar cipher was presented to an open class (in their language communications class) in a similar way. In both cases, most students showed interest in how encryption works and all students did the exercises. The discussion, after the demonstration, was very meaningful and several students were able to articulate ideas of enhancing the Caesar cipher code even after just one high level exposure to the topic.

## VI. CONCLUSION AND EVALUATION PLAN

This work presents the ongoing effort in building cyber security modules which can be used in a variety of courses at several levels – all years in a university bachelors program in computing, or at a community college associates program, or even in high school courses. The modules are being developed as a complete package which can be used independently or in a sequence. Each module package consists of module description and explanation, PowerPoint slides, lab exercise, lab solution and assessment exercises – which enables an external instructor to incorporate and use them in their courses without additional preparation. The modules outlined in this paper have been satisfactorily evaluated by the NSA evaluators, and upon completion of all modules, will be available to educators interested in cyber security education through their website.

The innovative approach of teaching cyber security across various computing curricula outlined in this paper will widen the pipeline to meet the demand of the nation for cyber security educated professionals.

#### ACKNOWLEDGEMENT

Partial support for this work was provided by the National Security Agency (NSA)'s grant project entitled "*Cyber Security Modules for Core, Major and Elective Courses in the Bachelor of Science (BS) Computer Science Curriculum.*"

#### REFERENCES

- [1] Akhtar Lodgher, Jeon Yang, "Cyber Security Modules for Core, Major and Elective Courses in the Bachelor of Science (BS) Computer Science Curriculum," NSA, Grant, Sep 2017-Aug 2018.
- [2] William Newhouse, Stephanie Keith, Benjamin Scribner, and Greg Witte, "National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework," <https://doi.org/10.6028/NIST.SP.800-181>, 2017.
- [3] The Joint Task Force on Computing Curricula (2013) Association for Computing Machinery (ACM) and IEEE-Computer Society, Computer Science Curricula 2013 Curriculum Guidelines for Undergraduate Degree Programs in Computer Science, Dec 2013. Retrieved from [https://www.acm.org/binaries/content/assets/education/cs2013\\_web\\_final.pdf](https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pdf).
- [4] Cybersecurity Curricula 2017, Curriculum Guidelines for Post-Secondary Degree Programs in Cybersecurity, Retrieved from [https://cybered.hosting.acm.org/wpcontent/uploads/2018/02/newcover\\_sec2017.pdf](https://cybered.hosting.acm.org/wpcontent/uploads/2018/02/newcover_sec2017.pdf).