Implementation of Security Modules with Model-Eliciting Activities in Computer Science Courses

Dr. Jeong Yang, Texas A&M University-San Antonio

Dr. Jeong Yang is an assistant professor in the Department of Computing and Cyber Security at Texas A&M University-San Antonio. She earned her Ph.D. in Computer Science and Software Engineering from Auburn University. Her current research interests include secure code analysis and visualization, and software security on various platforms such as stand-alone software systems and cloud-based or mobile applications. Her research also focuses on computing and cybersecurity education including the participation of women. She is an author or co-author of over 30 peer-reviewed journals and conference proceedings in these areas. She is a member of the IEEE Computer Society, ACM, ACM-W, Women in Cyber Security (WiCys), SWE(Society of Women Engineers), and NCWIT(National Center of Women in Technology).

Brandon Earwood, Texas A&M University-San Antonio
Dr. Young Rae Kim, Texas A&M University-San Antonio

Young Rae Kim, youngrae.kim@tamusa.edu, is an assistant professor of mathematics education in the Department of Curriculum and Instruction in the College of Education and Human Development at Texas A&M University-San Antonio. His research interests focus on how students' mathematical thinking develops and how teachers can use such thinking as a base for instruction. Storytelling and Model-Eliciting Activities (MEAs) are central to his recent research.

Dr. Akhtar Lodgher, Texas A&M University - San Antonio

Dr. Akhtar Lodgher is an A&M Regents Professor of Computing and Cyber Security and the Chair of the Department of Computing and Cyber Security (CCS) at Texas A&M University – San Antonio. His current research interests are in cyber security and software engineering. He has worked on several NSA and NSF funded grants for creating modules for infusing cyber security concepts into all core subjects of computer science education.

Implementation of Security Modules with Model-Eliciting Activities in Computer Science Courses

Abstract

Security is a critical aspect in the design, development, and testing of software systems. Due to the increasing need for security-related skills within software systems and engineering, there is a growing demand for these skills to be taught at the university level. A series of 41 security modules was developed to assess the impact of these modules on teaching critical cyber security topics to students. This paper presents the implementation and outcomes of the first set of six security modules in a Freshman level course. This set consists of five modules presented in lectures as well as a sixth module emphasizing encryption and decryption used as the semester project for the course. Each module is a collection of concepts related to cyber security. The individual cyber security concepts are presented with a general description of a security issue to avoid, sample code with the security issue written in the Java programming language, and a second version of the code with an effective solution. The set of these modules was implemented in Computer Science I during the Fall 2019 semester. Incorporating each of the concepts in these modules into lectures depends on both the topic covered and the approach to resolving the related security issue.

Students were introduced to computing concepts related to both the security issue and the appropriate solution to fully grasp the overall concept. After presenting the materials to students, continual review with students is also essential. This reviewal process requires exploring use-cases for the programming mechanisms presented as solutions to the security issues discussed. In addition to the security modules presented in lectures, students were given a hands-on approach to understanding the concepts through Model-Eliciting Activities (MEAs). MEAs are open-ended, problem-solving activities in which groups of three to four students work to solve realistic complex problems in a classroom setting. The semester project related to encryption and decryption was implemented into the course as an MEA.

To assess the effectiveness of incorporating security modules with the MEA project into the curriculum of Computer Science I, two sections of the course were used as a control group and a treatment group. The treatment group included the security modules in lectures and the MEA project while the control group did not. To measure the overall effectiveness of incorporating security modules with the MEA project, both the instructor's effectiveness as well as the student's attitudes and interest were measured. For instructors, the primary question to address was to what extent do instructors change their attitudes towards student learning and their teaching practices because of the implementation of cyber security modules through MEAs. For students, the primary question to address was how the inclusion of security modules with the MEA project improved their understanding of the course materials and their interests in computer science. After implementing security modules with the MEA project, students showed a better understanding of cyber security concepts and a greater interest in broader computer science concepts. The instructor's beliefs about teaching, learning, and assessment shifted from teacher-centered to student-centered, during his experience with the security modules and MEA.

1. Introduction

Software impacts a large number of people's lives in a myriad of ways. Software security is essential for guaranteeing that software is safe and behaves as intended. Markettos et al addressed that we face crises with security vulnerabilities in systems design of hardware, operating systems, and applications [1]. They advocated that security must be considered from the ground up in order to build complex hardware and software systems constructed for the new courses of vulnerabilities. Saydjari also discussed that engineers should be responsible for designing and building safe and secure systems and encourage them to do so in conjunction with system risk analysis and management [2, 3]. Yang et al pointed out that careless software design and implementations can cause a large number of vulnerabilities and attacks on the application itself. They stressed that security must be considered throughout the software development process. Toward secure software assurance, programming concepts must be taught to beginning programmers from a security perspective [4, 5]. This could be exercised through defensive secure programming, secure coding, and secure software development practices [5, 6]. A new knowledge area, Information Assurance and Security (IAS), and curricula were also established in order to better account for software security education at several universities [7].

In almost all universities, cybersecurity is taught as an "add-on" track or concentration where students take a series of courses related to cyber security in their junior and senior years. Students normally take basic computing core courses, and have the flexibility of choosing from several different tracks, such as gaming, software engineering, etc. Cyber security is so important that we believe that it is no longer a topic or a track – it is the way in which all software should be written. It no longer suffices to learn cyber security as an "add-on" towards the end, it should be taught in every course in a computer science curriculum because cyber security affects every major software component in any computing system.

A series of cyber security modules on various Computer Science (CS) topics have been developed for a National Security Agency (NSA) grant project [8]. The goal of the project is to teach cyber security concepts in CS courses from the first introductory course to senior level courses such as CS1, CS2, Secure Application Programming, Computer Security, Computer Network, Software Engineering I and II, and Cryptography. The objectives are to keep the modules complete and independent so that they can be easily integrated into the courses. Each module package consists of instructions, lab exercises and solutions, and assessment methods. The modules were also designed to incorporate the National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework (NCWF) topics of Cyber Threat and Vulnerabilities, Risk Management and Software Reverse Engineering [9].

The purpose of this paper is a) to describe a set of six security modules that was implemented in a Computer Science 1 course during the fall semester of 2019 at Texas A&M University-San Antonio and b) to report the results of evaluating teaching effectiveness of implementing the security modules with the Model-Eliciting Activity (MEA). The objective of including the security modules and the MEA project was to improve students' understanding of cyber security concepts as well as increase student interests in Computer Science.

2. Background

2.1 Cyber Security Modules

The first set of six cyber security modules was incorporated into the curriculum for a CS 1 course at Texas A&M University-San Antonio (A&M-SA). These modules were designed to introduce fundamental security concepts of defensive programming in beginners programming courses [5, 8]. They are currently available through the NSA public library for use: CLARK Cybersecurity Library [29]. The following six modules with ten lessons were included in the set of modules used.

Integer Errors (Module 1): The purpose of this module is to properly explain the difference between integer operations and floating-point operations in programming. Students are expected to understand the consequences of performing integer operations, such as integer division. These consequences include loss of precision and inaccurate results of operations. The proposed solution is to convert integer values to floating-point by using the cast operator when a floating-point operation would be more appropriate.

Securing Integer Boundary and Prevent Overflow (Module 2): Students should be familiar with the boundaries of the various integer data types and how to avoid overflow when performing math operations on integers. Students should also be aware of the asymmetry of these ranges. A solution to potential buffer overflows is to check the result of an operation to determine if the result can be stored in a variable of an integer data type or if an upcast must be performed.

Floating Point Inputs (Module 3 Lesson 1): This lesson emphasized checking a floating-point input to avoid any exceptional values, such as values that are too large or not a number. In either of these cases, the use of such values can lead to erroneous results or outputs. The solution for each of these cases requires using the isInifite() and isNaN() methods of the Double class. These methods return a boolean value that can be checked to determine if the input is safe to use.

Type Conversion (Module 3 Lesson 2): In some circumstances, a narrowing conversion must be performed on numeric data. This lesson focused on how to perform narrowing conversions safely and properly. The value being converted must be compared against the boundaries of the new data type before performing the conversion. For example, consider a value of type long being converted to type int. If the value is less than the maximum boundary and greater than the minimum boundary of the int type, the conversion can be performed.

Secure Variable Declarations (Module 4 Lesson 1): Secure variable declarations require clarity and consistency to avoid confusion. In this context, clarity and consistency means maintaining a clear, concise, and uniform approach to variable declarations and initializations. Declarations should be done on separate lines and initializations should be uniform across all variables. The practice of secure variable declarations also encourages proper commenting when necessary.

Scope of Variables (Module 4 Lesson 2): This lesson focused on minimizing variable scope in order to avoid programming errors. Variables should be applied to the smallest possible scope without losing functionality. This helps improve readability and maintainability. The purpose of variables is more clearly defined by using those variables in the minimal possible scope. Code

blocks also become more reusable as variable declarations are included rather needing to be redefined in new contexts.

Safe Division (Module 5 Lesson 1): The purpose of this lesson was to teach students to evaluate operands of division operators to avoid attempting to divide by zero. The proposed solution is to perform a check on the value used as the divisor in either division or modulo operations. The operation is only performed if this value is not zero.

Precision (Module 5 Lesson 2): This lesson focused on exploring the issue with storing decimal values in programs using floating-point data types. While these two data types are ideal for maintaining precision, there are cases where accuracy is a larger concern. Since numeric values are stored in binary, decimal portions of numbers cannot maintain perfect accuracy. A design approach to solving this problem is to consider the allowable tolerance of inaccuracies in a given application. One possible programming solution is to rely on classes available in Java that eliminate this issue, such as BigDecimal.

Bitwise and Arithmetic (Module 5 Lesson 3): This lesson emphasized the importance of keeping bitwise and arithmetic operations separate and distinct. Students must understand the consequences of attempting to use bitwise operations, such as bitwise shift left or right, to perform arithmetic operations. When performing either multiplication or division, students are encouraged not to use bitwise shifts as a logical equivalent since these operations do not offer identical behaviors.

Encrypting and Decrypting Text using Cipher (Module 6): This lesson is for students to have an understanding of basic encryption and decryption using a Caesar cipher. Caesar's encryption makes messages secret by shifting each letter three letters forward in the alphabet (sending the last three letters of the alphabet to the first three). Students understand how the cipher works, how to encode a simple alpha text and decode the encoded text using various Ciphers, describe the basic concept of how to break the cipher code, describe the code for hacking the Cipher.

2.2 Model-Eliciting Activities (MEA)

Model-Eliciting Activities (MEA) is an evidence-based teaching and learning methodology that supports the attraction and retention of underrepresented student populations in engineering, particularly female students [10, 11]. MEAs are open-ended, problem-solving activities in which groups of three to four students work to solve realistic complex problems [12] in a classroom setting. They were initially developed as research tools to explore students' thinking and procedures while collaboratively solving real-world problems. One of the important differences between MEAs versus typical engineering problem-solving activities involved in most course textbooks is the emphasis on multiple iterations of expressing, building, testing and revising conceptual models [13]. MEAs have been proven as an effective method to help engineering students become better problem solvers [14, 15, 16]. Other researchers have also documented MEA as a successful instructional tool for STEM education, particularly engineering education [17, 18].

A key feature of MEAs which makes them very suitable for this research is that MEAs are meant to be complementary materials for a curriculum, with the result that they can easily be integrated into existing curricula [16]. MEAs also have the potential of providing students with experiential

learning opportunities, based on engaging projects of the domain in which they are implemented [19] – computing and cyber security – for this project. Based on the fact that (a) MEA has been shown to support retention in engineering [10, 20], and (b) it helps students in becoming better problem solvers [14, 15, 16], this methodology was selected to assist in increasing the retention of students in computing and improve the learning of computing concepts for students.

During MEAs, students are required to develop or design mathematical/scientific/engineering tools or artifacts that an imaginary client needs to solve a realistic problem [12]. Student groups are usually given an article or video as an advanced organizer, introducing the realistic context and providing background information. After that, students individually answer readiness questions making students familiar with the practical context, and ready to engage in the problem task. A problem statement is provided for the students that may specify the client's requirements. Students are also given enough information, without more background research, to create solutions addressing the needs of their client. Students work in small groups of three to four to express and develop alternative solutions - and choose the best one. They design and build it as a prototype. Then they test and revise it to meet the needs of their client successfully. Finally, student groups present their solutions and ideas in the whole class, and they are given time for self-reflection and final revision of their models.

3. Implementation of Security Modules with Model-Eliciting Activities

3.1 Incorporation of Cyber Security Modules

For each of the 9 lessons introduced in the CS 1 course, an explanation is provided of how that lesson was incorporated into the course curriculum. Table 1 presents the lessons and the MEA project in relation to the chapter of the textbook that is covered at the time that lesson is introduced. The book used for the course was Starting Out With Java: From Control Structures through Objects, 7th Edition.

Table 1. Cyber Security Modules with Lessons and Chapters to Cover Modules.

Chapter to Cover Module	Module#. Lesson(s)	Implementation Approach
Ch. 2. Java Fundamentals	4.1 Secure Variable Declarations	Understanding how to write secure variable declarations is critical to producing overall secure code.
Ch. 2. Java Fundamentals	1 Integer Errors	Introduced with arithmetic operations. Students must be made familiar with how integer/floating-point division is handled.
Ch. 3. Decision Structures	5.1 Secure Division	Similar to type conversion, while dividing by zero is primarily a topic to be addressed with arithmetic operations, the solution to security risk requires that students know conditional logic.
Ch. 3. Decision Structures	2 Securing Integer Boundaries & Prevent Overflow	Introduced with arithmetic operations. Students should understand the limits of numeric data types and what happens when those limits are reached and exceeded to avoid integer overflow and understand why it happens.
Ch. 3. Decision Structures	5.2 Precision	Introduced after students have been exposed to the Scanner class in chapter two of the textbook. The combination of seeing arithmetic operations, type conversions, and import

		statements using another class provides students with the tools necessary to fully understand using the BigDecimal class as a solution to precision errors in programs.
Ch. 4. Loops and Files	4.2 Scope of Variables	Covered after teaching students about loops. Many of the examples presented in this lesson deal with declaring the variable used as a loop counter outside of the loop.
Ch. 4. Loops and Files	3.1 Floating Point Inputs	Introduced after teaching students about conditional logic and booleans, which are covered in chapter 3 of the textbook. This helps students understand the output from testing values using the isInifite() and isNaN() methods as well as how to construct programming structures that will execute some code if the values given are valid.
Ch. 4. Loops and Files	3.2 Type Conversion	Even though this lesson is primarily dedicated to type conversions, students must also be familiar with conditional logic. This knowledge is crucial for students to understand how to check a value against the range of a data type.
Ch. 7. Arrays and the ArrayList Class	5.3 Bitwise Arithmetic	Covered alongside arrays. Bit operations are not included with the normal course materials, so trying to determine the most effective time to expose students to secure bitwise operations became a matter of preference.
Ch. 4. Loops and Files Ch. 5. Methods Ch. 7. Arrays and the ArrayList Class	6 Caesar Cipher – Encryption and Decryption: MEA Semester Project	To properly perform the Caesar cipher on a block of text, students should be familiar with how to construct loops to iterate over each character. Presenting this an array of characters effectively describes this iterative approach. Students should also understand methods because a separate encryption and decryption method are used in an example program.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

1. **Caesar Cipher**

   Encrypt: Caesar encryption makes messages secret by shifting each letter three letters forward in the alphabet (sending the last three letters of the alphabet to the first three). It processes mathematically, first replace each letter by an element of Z_{26}, that is, an integer from 0 to 25 equal to one less than its position in the alphabet as shown in the Figure above.

   This can be represented by a function f: **f(p) = (p + 3) mod 26**, where p represents a letter.

   In the encrypted version of the message, the letter represented by p, is replaced with the letter represented by (p + 3) mod 26.

Figure 1. Individual Activity - Caesar Cipher.

3.2 Development of Model-Eliciting Activities

To incorporate the teaching of cyber security concepts of the modules, the development of effective MEAs is designed by six design principles: Reality, Model Construction, Model

Documentation, Self-Assessment, Generalizability, and Effective Prototype [3]. The MEA project implemented for this study involved students studying three simple encryption algorithms in an individual activity. Students learned about the Caesar cipher, affine cipher, and block cipher and answered a set of questions related to each algorithm to demonstrate their understanding. Figure 1 presents a section of the individual activity.

To follow up on the individual activity, student groups were then given the task of designing their unique encryption algorithm based on the fundamental principles learned. Students were presented some background information about a problem requiring the design of a new encryption algorithm. Students were expected to use the knowledge gained from the individual activity to design an entirely new algorithm within groups of three. **Appendix A** gives the background description for the group activity. Student groups prepared both a written description, either as pseudocode or step by step instructions, of their algorithm as well as a visual description, either as a diagram or flowchart. On the following lecture day, students presented the solutions to the rest of the class. To make expectations of group presentations clear, students were provided with a rubric for grading peer presentations during the day to work on the group activity. The peer evaluation rubric is attached in **Appendix B**.

3.3 Group Project Assessment

For each of the groups involved in the MEA project, there were three typical outcomes: combining the previously learned algorithms, using a combination of previously learned algorithms and additional algorithms not covered in the individual assignment, or attempting to produce entirely new algorithms that do not already exist. While this order does highlight the least to most inventive approaches to solving the problem, this order also represents the most to least practical ideas. The most inventive ideas presented make for interesting design approaches but would prove to be impractical or infeasible in implementation. Table 2 lists the groups in order of presentation, the algorithms involved in that group's proposed solution, and the average score that group received from peer reviewed grading using the presentation rubric from Appendix B.

Table 2. Group Activity Solutions and Presentation Scores.

Group	Solution	Presentation Score
1	Affine cipher using different values for split alphabet and block cipher	14
2	Sequence of keys needed to open successive messages, segmented message applying multiple encryption algorithms, and salting	18.7
3	Vigenere, affine cipher, and transposition	17.5
4	Reflection method (shifting letters by splitting alphabet)	15.7
5	Block cipher, conversion to numeric values, and hashing	17.3
6	Caesar cipher replacing letters with special characters	16.3
7	Block cipher with swapped blocks and Caesar cipher	15.3
8	Salting by adding letters of the alphabet and affine cipher	17.5

4 Experimental Design

4.1 Research Questions

To study the teaching effectiveness of implementing security modules with MEAs in CS courses, the nature of the intervention (design) and the results of student learning were investigated to improve the design of the intervention by using the design experiment methodology [11]. This methodology allows the investigation of how an intervention affects student learning and teaching practices in a complex learning environment [21]. The research also follows the basis of the methodology of Research Type #5 – Effectiveness Research [22] and it has two parts: instructor effectiveness and students' attitudes and interest.

The studies of instructor effectiveness and students' attitudes and interest were guided by the following research questions:

1. To what extent do instructors change their attitudes towards student learning and their teaching practices because of the implementation of cyber security modules through MEAs?
2. In what ways do the implementation of cyber security modules through MEAs change students' attitudes towards learning in computer science?
3. In what ways do the implementation of cyber security modules through MEAs enhance students' interest in computer science?

Table 3. Pre- and Post- Beliefs Interview Protocol Questions.

Pre-Interview		**Post-Interview Question**
Question	***Category***	
1. How do you describe your role as the instructor?	Teaching practice	1. What are some changes in your classrooms after the use of MEAs for cyber security modules? 2. What are some differences between your expectation and your observation in the student work through the use of MEAs for cyber security modules?
2. How do your students best learn engineering?	Student learning	
3. How do you maximize student learning in your classroom?	Teaching practice	
4. How do you know when your students understand?	Assessment	
5. How do you decide what to teach or what not to teach?	Teaching practice	
6. How do you decide when to move on to a new topic in your class?	Assessment	
7. How do you know when learning is occurring in your classroom?	Student learning	

4.2 Participants

To answer the research questions, an experimental study was also conducted while the security modules were incorporated into the classroom teaching in the CS 1 course at A&M-SA. 52

undergraduates and one instructor participated in this study. The participation was voluntary. The instructor taught two sections of the course: one with the implementation of the cyber security modules with the MEA project (treatment group) and another without the implementation (control group). Both sections had 26 students enrolled.

4.3 *Data Collection and Analysis*

Both quantitative and qualitative data were collected for the study through semi-structured pre- (beginning of the semester) and post-interviews of the instructor, observation of the implementation of the cyber security modules with the MEA, student outcomes on the MEA, and open-response student surveys from both treatment and control groups at the end of semester. The pre- and post-interview protocol for the instructor included seven questions, which were modified and adapted from previous studies [23, 24]. This is to access instructors' current views on instructional practices, student learning, and student understanding. Additional questions were asked for the post-interview to assess instructor views on the implementations of the cyber security modules with the MEA. For each interview, extensive field notes were taken. Table 3 shows seven pre-interview questions categorized into teaching practice, student learning, and assessment, and two post-interview questions. The observation instrument of instructor implementation of the security modules and MEA consisted of the researchers' field notes and the instructor's interaction with students. The student outcomes on the MEA were student group reports that include their solutions, processes, and group presentations. Finally, the open-response student survey included four questions for both treatment and control groups to explore student learning experience, and an additional question for the treatment group to examine the effectiveness of the implementation of the modules with the MEA (Table 4).

Table 4. Open-Response Survey on Student Learning Experience.

Question	Involved Groups
1. How likely are you to enroll in the [Next Course in the Computer Science Sequence] next semester?	Both Treatment and Control Groups
2. Explain briefly what helps you learn in the Computer Science courses at your institution, preferably by using an example.	Both
3. What changes, if any, would you suggest to make the courses more helpful?	Both
4. Have you become more competent due to participation in the courses?	Both
5. Do the cyber security modules and MEAs contribute to your interest and understanding of computer science?	Treatment Group

The interview field notes, and student survey responses were analyzed by both deductive and inductive approaches to coding the qualitative data [25, 26]. First, the field notes for the interview questions were coded by two researchers based on preset rubrics that were adopted from previous studies [23, 24, 27]. The rubrics for each of the first seven questions in Table 3 consists of five categories ranging from teacher-centered to more student-centered beliefs: (1) Traditional, (2) Instructive, (3) Transitional, (4) Emerging Constructivist, and (5) Experienced Constructivist. More teacher-centered beliefs are coded (1) for traditional, "which indicates beliefs that teachers are providers of knowledge," and (2) for instructive, "which indicates beliefs

that students should have experiences that mimic the teacher or are closely monitored and directed by the teacher." Transitional coded (3) indicates "beliefs that instruction should be teacher-led but have student input." More student-centered beliefs were coded (4) for emerging constructivist and (5) for constructivist beliefs [23].

Second, the two researchers formulated codes (categories) from student survey responses to questions 1 and 2 in Table 4 as they became apparent from the data. The two researchers reached a consensus on the codes throughout the coding process. The codes for the question 1 were five Likert Scale of interests: (1) Not likely, (2) Possibly, (3) Likely, (4) Very Likely, and (5) Definitely. The codes for the question 2 were (1) "not sure"; (2) "student-centered" strategies (e.g., hands-on, by doing, collaborative, interactive); (3) "neutral" (e.g., assignments, repetition); and (4) "teacher-centered" strategies (e.g., detailed instructions; PPT slides; textbook). Once the coding schemes were established, the two researchers applied the codes to the student responses to the questions. There were missing responses from some students to each question and they were not included in the data analysis process. Thus, the total numbers of student responses were different from each question. In coding the data by the two researchers, Cohen's K coefficient of the inter-rater agreement was 0.91, indicating an acceptable level of reliability [26]. The two researchers also discussed differences in coding and made a consensus on the coding discrepancies.

4 Results and Discussion

5.2 Instructor Change in Beliefs Over the Semester

The seven main interview questions were coded for the pre- and post-interviews to explore the change in instructor beliefs as presented above: Traditional (1), Instructive (2), Transitional (3), Emerging Constructivist (4), and Experienced Constructivist (5) (see Table 5).

Table 5. Faculty Beliefs of Teaching, Learning, Assessment.

Interviews		First (at the beginning of the semester)	Second (at the end of the semester)
Teaching	Role as Instructor	(1)	(3)
	Maximize Student Learning	(2)	(2)
	What to Teach	(1)	(1)
Learning	How Students Learn Best	(2)	(3)
	Learning Occurs	(3)	(3)
Assessment	When Students Understand	(3)	(5)
	When to Move on	(1)	(2)

As shown in Table 5, the instructor indicated that he was an instructor who displayed a combination of "traditional" and "instructive" traits at the beginning of the semester. In his first interview, the instructor viewed his role as a teacher "to cover general concepts" (traditional). He taught "what the industry standards are" and "would focus on generalizing" (traditional). In order to maximize students learning, he provided students with "real-world examples or demonstrations" (instructive). He believed that students best learn to engineer when they are given opportunities for "practicing a lot" (instructive). He knew when learning was occurring in

his students by “asking questions” (transitional). The instructor also tried to measure student understanding through “holding a conversation about the topics being discussed” with students during the lecture (transitional). However, he decided when to move on to a new topic based on “how much time spent on each topic” (traditional). After the implementation of the cyber security modules with the MEA project, the instructor described his job as an instructor “to give them [students] information outside of the exams and labs to help them understand the materials” (transitional). His emphasis on student understanding is a meaningful change in his beliefs on teaching towards a more student-centered view. However, he focused on “mostly cover[ing] topics in the book” when deciding what to teach (traditional). He believed that he could maximize student learning by “Constantly asking questions” and “having them[students] step through a program (design + implementation)” (instructive). “Engaged by asking questions or asking him to repeat what they [students] didn’t understand,” the instructor knew whether learning was occurring in his classes (instructive). He felt that students learn best “from feedback” and “from smaller, more controlled hands-on projected” (transitional). His emphasis on the use of hands-on activities is a positive change in his beliefs on student learning. His beliefs about assessment also changed to more student-centered views. He decided whether to move to the next topic in a class by considering the needs of students: “If students are still struggling on a topic, then he’ll stick with it for a while longer” (instructive). In order to assess student understanding, he considered if his students could “reciprocate and ask questions beyond course materials” (experienced constructivist).

In addition, as shown in Table 6, a graphical representation was created to visualize the instructor’s shift in overall belief system over the semester as Moore et al. (2015) did.

Table 6. Instructor Change of Beliefs over the Semester.

Interviews	Traditional (1)	Instructive (2)	Transitional (3)	Emerging (4)	Constructivist (5)
1st Interview	***	**	**		
2nd Interview	*	**	***		*

* Each asterisk represents the code the answer received for one of the seven interview questions.

Table 6 shows a general shift of responses to the right throughout the study. This indicates that the instructor exhibited a shift in his beliefs towards a more student-centered view using MEAs for the cyber security modules, even though it is a short one-semester period. The instructor also shifted from an instructor who displayed a combination of “traditional” and “instructive” traits to a more student-centered instructor having “instructive,” "transitional," and “constructivist” views. This indicates that the instructor’s beliefs about teaching, learning, and assessment shifted from teacher-centered to student-centered, during his experience with the MEA. This meaningful finding answered the research question #1. Even though he mentioned that “[it] didn’t impact him too much since his view of teaching aligned with the use of MEAs,” his responses to the last two questions regarding implementations of the cyber security modules with the MEA supported a positive impact of the use of MEAs on his beliefs and decisions about teaching, learning, and assessment: “*The project is styled differently & topics are covered differently*”; “*The MEA projects, most students exceeded beyond [my] expectation*”, and “[I] liked the practicality of the modules and MEA project.”

5.3 Student Experience with MEAs

The student survey responses to the first question in Table 4 were coded by the five Likert Scale of interests: (1) Not likely, (2) Possibly, (3) Likely, (4) Very Likely, and (5) Definitely, while the responses for the second question were coded by the four categories: (1) "not sure"; (2) "student-centered" strategies; (3) "neutral"; and (4) "teacher-centered" strategies. The responses for these two questions were explored to indirectly examine the impact of the use of MEA project for the cyber security modules on student interest and understanding of learning computer science, along with the direct question # 7 for the treatment group – *Do the cyber security modules and MEAs contribute to your interest and understanding of computer science?*

Table 6. Likelihood of Taking a Next Computer Science Course

Group	Not Likely	Possibly	Likely	Very Likely	Definitely	Total
Treatment	1 (4.8%)	1 (4.8%)		14 (66.7%)	5 (23.8%)	21
Control	3 (13.0%)	1 (4.3%)	4 (17.4%)	12 (52.2%)	3 (13.0%)	23

Table 6 summarize data at the five Likert Scale showing the interest of enrolling in a CS course next semester. 90.5 % (Very Likely and Definitely: 19/21) of the participants in the treatment group wanted to enroll in the next CS course in a sequence. Only (2/21) 4.8 % of them said it was unlikely for them to take the next course. In the control group, 65.2% (15/21) of participants wanted to take the next course in the next semester, and 13% (4/23) of them didn't want to take the next course in the computer science course. Although the frequency counts and percentages are not statistically analyzed due to the low sample size, this finding is still useful to explore general patterns in the data [24]. The difference between the two groups in the likelihood can support a positive impact on student interests in computer science using MEAs.

Table 7. Circumstances that Helped Students Learn in the Computer Science Course.

Group	Not sure	Teaching-centered	Neutral	Learning-centered	Total
Treatment		7 (28%)	9 (36%)	9 (36%)	25
Control	1 (4.8%)	6 (28.6%)	7 (33.3%)	7 (33.3%)	21

Table 7 shows the results of the second question in Table 4 that is related to circumstances that helped students learn CS concepts. There was no significant difference between treatment and control groups. However, the data shows that many participants in both groups did learn from learning-centered environments. Then they thought they learned the concepts better in hands-on activities, group activities, or real-world problem solving, which are the main characteristics of MEAs. This could also support that the use of the MEAs on teaching cyber security modules enhance students' interest and give them a better understanding of computer science.

For the question #5 in Table 4, "*Do the cyber security modules and MEAs contribute to your interest and understanding of computer science?"*, approximately 81 % (17/21) of the students in the treatment group appreciated that the use of the cyber security modules with the MEA project contributed to their interests and understanding of computer science.

In Addition, 5 students from the treatment group suggested more hands-on activities, group activities, or real-world problems to make the course more helpful. These responses indirectly reflect their experience with the MEA. In the meantime, only one student from the control group suggested more real-world problems. This difference could also support the contribution of the implementation of the cyber security modules with the MEA project to students' interest and understanding of computer science. In summary, students have experienced that the MEA project in the corporation of the cyber security modules in the course enhances their interests and attitude toward learning in computer science. This result supports the research questions 2 and 3.

5 Conclusion and Future Works

This paper presents the outcomes of implementing six security modules in a Freshman level course at A&M-SA during the Fall 2019 semester. Students in the course were introduced to computing concepts related to both the security issue and the appropriate solution to fully grasp the overall concept. In addition to the security modules presented in lectures, students were given a hands-on approach to understanding the concepts through Model-Eliciting Activities (MEAs). The semester project related to encryption and decryption was implemented into the course as an MEA project. To assess the effectiveness of incorporating security modules with the MEA project, two sections of the course were used as a control group and a treatment group. The treatment group included the security modules in lectures and the MEA project while the control group did not. To measure the overall effectiveness of incorporating security modules with the MEA project, both the instructor's effectiveness, as well as the student's attitudes and interest, were investigated. For instructors, the primary question to address was to what extent do instructors change their attitudes towards student learning and their teaching practices because of the implementation of cyber security modules through MEAs. For students, the primary question to address was how the inclusion of security modules with the MEA project improved their understanding of the course materials and their rate of interest in computer science. After implementing security modules with the MEA project, students showed enhanced interests and attitudes toward learning in computer science. The instructor's beliefs about teaching, learning, and assessment shifted from teacher-centered to student-centered, during his experience with the cyber security modules and MEAs.

In order to further investigate and analyze the effectiveness of security modules in computer science disciplines, the sets of remaining modules with other MEA projects will be presented to students in more Computer Science courses, such as Computer Science 2, Discrete Structures for Computing, Computer Networks, Computer Security, Software Engineering, and Cryptography in future semesters at more universities. The study continues to evaluate the teaching effectiveness of the remaining secure modules associated with the MEAs.

Acknowledgment

Partial support for this work was provided by the National Science Foundation (NSF)'s grant project entitled "Recruiting and Retaining Students into Computing" under the award #1832433. This material is based upon work supported by the Grant**.** Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Markettos, A. T., Watson, R. N. M., Moore, S. W., Sewell, P., & Neumann, P. G. (2019). Through Computer Architecture, Darkly, Communications of the ACM, Vol. 62 No. 6, Pages 25-27, 10.1145/332528.

[2] Saydjari, O. Sami. (2019). Engineering Trustworthy Systems: A Principled Approach to Cybersecurity. Communications of the ACM, Vol. 62 No. 6, Pages 63-69, 10.1145/3282487.

[3] Stamat, M. L. & Humphries, J. W. (2009). Training ≠ Educating Secure Software Engineering Back in the Classroom. WCCCE '09 May 1-2, 2009, Burnaby, BC, Canada. ACM 978-1-60558-415-7.

[4] Yang, J., Lodgher, A., & Lee, Y. (2018). Secure Modules for Undergraduate Software Engineering Courses. *2018 IEEE Frontiers in Education Conference (FIE)*, San Jose, CA, USA, doi: 10.1109/FIE.2018.8658433.

[5] Yang, J. & Lodgher, A. (2019). Fundamental Defensive Programming Practices with Secure Coding Modules. *2019 International Conference on Security and Management*, Las Vegas, NV.

[6] Yuan, Xiaohong; Yang, Li; Jones, Bilan; Yu, Huiming; & Chu, Bei-Tseng. (2016) "Secure Software Engineering Education: Knowledge Area, Curriculum and Resources," *Journal of Cybersecurity Education, Research and Practice*: Vol. 2016: No. 1, Article 3.

[7] 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.

[8] Lodgher, A and Yang J, 2017, Cyber Security Modules for Core, Major and Elective Courses in the Bachelor of Science (BS) Computer Science Curriculum, NSA Grant, Sept 2017-Aug 2018.

[9] William Newhouse, Stephanie Keith, Benjamin Scribner, Greg Witte, "National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework," National Institute of Standards and Technology (NIST), U.S. Department of Commerce, 2017, https://doi.org/10.6028/NIST.SP.800-181.

[10] Diefes-Dux, Heidi, Deborah Follman, P.K. Imbrie, Judith Zawojewski, Brenda Capobianco, Margret Hjalmarson. (2004). "Model Eliciting Activities: An In-class Approach to Improving Interest and Persistence of Women in Engineering," Proceedings of the 2004 American Society for Engineering Education Annual Conference and Exposition, 2004.

[11] A. Collins, D. Joseph, and K., "Design Research: Theoretical and Methodological Issues, The Journal of the Learning Science,"13(1), 15-42, 2004.

[12] Lesh, R., & Doerr, H. M. (2003). Foundations of a models and modeling perspective on mathematics teaching, learning, and problem solving. In R. Lesh & H. M. Doerr (Eds.), Beyond constructivism: Models and modeling perspectives on mathematics problem solving, learning, and teaching (pp. 3-33). Mahwah, NJ: Lawrence Erlbaum Associates.

[13] Lesh, R., Hoover, M., Hole, B., Kelly, A., & Post, T. (2000). Principles for developing thought-revealing activities for students and teachers. In A. Kelly & R. Lesh (Eds.), Research design in mathematics and science education (pp. 591-646). Mahwah, NJ: Lawrence Erlbaum and Associates.

[14] Frank, B., & Kaupp, J. (2012). Evaluating integrative model eliciting activities in first year engineering. Proceedings of the 2012 Canadian Engineering Education Association Conference. Retrieved from http://www.academia.edu/2761700/Evaluating_Integrative_Model_Eliciting_Activities_in_First_Year_Engineering.

[15] Moore, T. J., Miller, R. L., Lesh, R. A., Stohlmann, M. S., & Kim, Y. R. (2013). Modeling in engineering: The role of representational fluency in students' conceptual understanding. Journal of Engineering Education, 102(1), 141-178.

[16] Kaupp, J., Frank, B., & Chen, A. (2014). Evaluating critical thinking and problem solving in large classes: Model eliciting activities for critical thinking development. Toronto, Canada: Higher Education Quality Council of Ontario. Retrieved from http://www.heqco.ca/Site Collection Documents/Formatted%20Queen%27s_Frank.pdf.

[17] Hamilton, Lesh, Lester, Brilleslyper. (2008). Model-Eliciting Activities as bridge between engineering education research and mathematics education research, ASEE.

[18] Moore, T. J., Guzey, S. S., Roehrig, G. H., Stohlmann, M., Park, M. S., Kim, Y. R., Callender, H. L., & Teo, H. J. (2015). Changes in faculty members' instructional beliefs while implementing model-eliciting activities. Journal of Engineering Education, 104(3), 279-302.

[19] Lesh, R., & Yoon, C. (2004). Evolving communities of mind - in which development involves several interacting and simultaneously developing strands. Mathematical Thinking and Learning, 6 (2), pp. 205-226.

[20] Brown, H. A., Forde, Tomothy. (2006). Addressing Diversity in schools:Culturally Responsive Pedagogy, Practitioner Brief Series.

[21] Nuñez, A.-M. (2015). "Hispanic-Serving Institutions: Where are they now?" A commissioned paper presented at the meeting "Hispanic-Serving Institutions in the 21st century: A convening" at the University of Texas El Paso. El Paso, TX.

[22] Common Guidelines for Education research and Development, A Report from the Institute of Education Sciences, U.S. Department of Education, and the National Science Foundation, 2013.

[23] Moore, T. J., Guzey, S. S., Roehrig, G. H., Stohlmann, M., Park, M. S., Kim, Y. R., Callender, H. L., & Teo, H. J. (2015). Changes in faculty members' instructional beliefs while implementing model-eliciting activities. Journal of Engineering Education, 104(3), 279-302.

[24] Roehrig, G. H. & Luft, J. A. (2004). Inquiry teaching in high school chemistry classrooms: The role of knowledge and beliefs. Journal of Chemical Education, 81(10), 1510-1516.

[25] Corbin, J., & Strauss, A. (2008). Basics of qualitative research (3rd ed). Thousand Oaks, CA: Sage.

[26] Miles, M. B., & Huberman, A. M. (1994). Qualitative data analysis. Thousand Oaks, CA: Sage.

[27] Luft, J. A., Bang, E. J., & Roehrig, G. H. (2007). Supporting beginning science teachers. The Science Teacher, 74(5), 24-29.

[28] Altman, D. G. (1991). Practical statistics for medical research. London, United Kingdom: Chapman and Hall.

[29] CLARK Cybersecurity Library, Retrived from https://clark.center/home.

Appendix A: Group Project Activity

Model Eliciting Activity (MEA) – Cipher Algorithm

TO: CSCI 1336 Programming Fundamentals I

FROM: Earwood, Brandon

Lecturer of Computing and Cyber Security, Texas A&M University-San Antonio

SUBJECT: Fwd: Re: Re: Cipher Competition

The Department of Computing and Cyber Security is hosting a competition and has invited our class to help them create a cipher algorithm. The competition rules are to design a cipher algorithm extending from the various Cipher algorithms, but they have included a scenario where a hacker has intercepted both plaintext and cipher text messages from your algorithm by using a Man-In-The-Middle Attack, where attackers can alter the communication data between two parties.

As a cyber developer, you will be required to use a wide array of tools and techniques to design and verify your cipher algorithm. One of the most important skills in cryptography is encryption and decryption. By understanding how these processes work, a developer can manipulate these functions to perform more complex standards for encryption algorithms.

In line with this view, the department requests aid to develop a new algorithm. This is where your team comes in. Your team needs to design a more secure version of the cipher algorithm. The design should be reusable for encrypting and decrypting any message or any set of messages. You are also expected to use the concepts that you have learned in class to handle type conversion of data types as well as catch exceptions and unexpected behaviors.

Your team will submit detailed documentation on how your cipher algorithm is designed with reasons behind the decisions you made for the algorithm. For the documentation, you are required to create a diagram or flowchart and provide pseudo code or step by step instructions of the algorithm showing each step, as well as the documentation explaining how it is working in encryption and decryption with a plaintext message. Use the phrases “Hello” and “Security” as plaintext messages to demonstrate the encryption and decryption process. This requires that your team gives a detailed explanation on how messages are being converted from plaintext message to cipher text and vice versa.

Please let me know if you have any questions.

Earwood.

Begin forwarded message:

> From: Department of Computing and Cyber Security, Texas A&M University-San Antonio

> To: Jeong Yang, Brandon Earwood

> Subject: Re: Cipher Competition

> Thank you for agreeing to collaborate with us in this competition. We are sure that the skills demonstrated in this competition will be truly beneficial to both parties. Most of all, your project with the department could significantly contribute to the technological workforce and regional economy by establishing strong ties with students and cryptography in the cyber security community.

>

> On behalf of the department, we would again like to express our appreciation for all the support you and your students will provide.

Appendix B: Group Presentation Rubric.

	Noteworthy	Acceptable	Needs minor revisions	Needs major revision	Needs redirection
Needs of the Client: Does the solution meet the needs of the client?	The tool not only works for the immediate situation, but it also would be easy for others to modify and use it in similar situations.	No changes will be needed to meet the immediate needs of the client.	The product is nearly ready to be used. It still needs a few small modifications, additions, or refinements.	The product is a good start toward meeting the client's needs, but a lot more work is needed to respond to all of the issues.	The product is on the wrong track. Working longer or harder won't work.
Documentation: Does the documentation completely explain the procedure used to arrive at the solution?	The documentation provides enough detail for the client to implement the suggested solution, and it includes information about how to alter the solution for different but similar circumstances.	The documentation provides enough detail for the client to implement the suggested solution without additions or clarification.	The documentation provides enough detail that the client could implement the procedure with only minor clarification.	The documentation only describes the solution process generally. The client would be unable to implement the solution process simply from the information provided in the documentation. The client would need clarification, more information, or help.	The documentation describes very little of the solution process.
Presentation: Was the information shared in a professional manner and communicated clearly?	The presentation was conducted in a professional and creative manner. The audience clearly understood the solution process and could relate it to other similar situations. The presentation was well planned, organized, complete, all group members participated in the presentation, and visual aids were used.	The presentation was conducted in a professional manner. The audience understood the solution process. Presentation was well planned and organized, complete, and visual aids were used. All members participated in the presentation.	The presentation was conducted in a professional manner. The audience understood most of the solution process. Little clarification was needed.	The presentation was vague and/or unorganized. The audience only partially understood the solution process.	The presenters were unprepared for the presentation. The presentation was unorganized, unprofessional, and contained no use of visual aids.

How would you rate this algorithm's level of security? (Select one of the following from a 1-5 scale.)

Not Secure	Somewhat Secure	Average Security	Sufficiently Secure	Strongly Secure
1	2	3	4	5