Curriculum Development for Embedded Systems Security

Dr. Janusz Zalewski, Florida Gulf Coast University

Janusz Zalewski, Ph.D., is a professor of computer science and software engineering at Florida Gulf Coast University. Prior to an academic appointment, he worked for various nuclear research institutions, including the Data Acquisition Group of Superconducting Super Collider and Computer Safety and Reliability Center at Lawrence Livermore National Laboratory. He also worked on projects and consulted for a number of private companies, including Lockheed Martin, Harris, and Boeing. Zalewski served as a chairman of the International Federation for Information Processing Working Group 5.4 on Industrial Software Quality, and of an International Federation of Automatic Control Technical Committee on Safety of Computer Control Systems.His major research interests include safety related, real-time embedded and cyberphysical computer systems, and computing education.

Dr. Andrew J Kornecki, Embry-Riddle Aeronautical Univ., Daytona Beach

Andrew J. Kornecki, Ph.D., is a professor at the Department of Electrical, Computer, Software and Systems Engineering, at Embry Riddle Aeronautical University. He has more than 30 years of research and teaching experience in areas of simulation, safety, and real-time computer systems. He contributed to research on intelligent simulation training, safety-critical software, was engaged in work on certification issues and assessment of tools for real-time, safety-critical systems, and served as a visiting researcher with the Federal Aviation Administration (FAA). He conducted industrial training on real-time, safety-critical software in medical and aviation industries and for the FAA Certification Services.

Dr. Bogdan Denny Czejdo, Fayetteville State University

Bogdan Denny Czejdo is a Belk Distinguished Professor of Science and Technology in the Department of Mathematics and Computer Science, at Fayetteville State University. He received the M.S. and the Ph.D. degree from Warsaw University of Technology. His research focuses on visual languages, databases, knowledge bases, and robotics. He received funding for many projects in these areas and has more than 150 research publications. He has served on several editorial boards for professional journals and on a number of program committees for prestigious conferences. He has taught a wide range of computer science courses.

Dr. Fernando Garcia Gonzalez, Florida Golf Coast University

Dr. Fernando Gonzalez joined FGCU as an Assistant Professor in the Computer Engineering Program in the fall of 2013. Previously he was an Assistant Professor within the Engineering, Math, and Physics Department at Texas A&M International University in Laredo, Texas. Prior to that he was a Technical Staff Member (researcher) for the U.S. Department of Energy at Los Alamos National Laboratory in Los Alamos, New Mexico. Dr. Gonzalez was also a faculty member in the Electrical and Computer Engineering Department of the University of Central Florida. Dr. Gonzalez graduated from the University of Illinois in 1997 with a Ph.D. in Electrical Engineering. He received his Master's degree in Electrical Engineering and his Bachelor's degree in Computer Science from Florida International University in 1992 and 1989. Dr. Gonzalez research interest includes the intelligent control of large scale autonomous systems, autonomous vehicles, discrete-event modeling and simulation and human signature verification.

Dr. Nary Subramanian, University of Texas at Tyler

Nary (Narayanan) Subramanian is currently an Associate Professor of Computer Science at The University of Texas at Tyler, Tyler, Texas. Dr. Subramanian received his Ph.D. in Computer Science from The University of Texas at Dallas. His specialization is software engineering with particular focus on software architectures and requirements engineering. He co-founded the International Workshop on System/Software Architectures (IWSSA) and served as a co-chair for seven years between 2002 and 2011. He established and directed the Center for Petroleum Security Research at UT Tyler. He has over fifteen years' experience in industry in engineering, sales, and management. He is a member of the IEEE. His research interests include software engineering, system engineering, and security engineering.

Dr. Dawid Trawczynski, Advanced Micro Devices

Dr. Dawid Trawczynski is a senior design engineer at Advanced Micro Devices, in Orlando, Florida. He graduated with MSc in Computer Engineering from the University of Central Florida, in 2003, and completed his PhD in Computer Science, at Warsaw University of Technology, in 2009. His research interests include testing and verification methodologies, as well as embedded software engineering, dependability and reliability analysis of embedded systems, and security of cyberpysical systems, with emphasis on time-triggered applications.

Curriculum Development for Embedded Systems Security

Abstract

The paper describes issues involved in the development of a modular curriculum for embedded systems security. The topics selected in the first round include: (1) General educational modules on security, such as "General Introduction to Computer Security", "Introduction to Cryptography" and "Embedded Systems Security"; (2) Security of specific technologies used in embedded systems, which in this edition involve "FPGA Security", "RFID Security" and "SCADA Security"; and (3) Software aspects of embedded systems security, such as "Java Security" and "Threat Modeling."

Each module includes the following components: Objectives, Introduction, Student Activities composed of Suggested Readings and Hands-on Exercise, and Assessment. Eight modules developed thus far have been tested in undergraduate courses on embedded systems and computer networks at one academic institution, and are currently being revised for testing at other universities across the nation. They are all available on the Internet and are being expanded to cover aspects specific to cyberphysical systems security.

Introduction

Security of embedded systems is of primary concern from the point of view of the nation's economy and safety of its citizens. According to some estimates, the number of embedded devices in use is about two orders of magnitude higher than that of desktops. Moreover, as the Chief Scientist of the U.S. Air Force recently stated at the 2013 CSIIR Workshop[1)], in his estimate by the year 2025 there will be seven trillion IP enabled devices in existence, all forming a humongous ecosystem that would need a well-educated workforce. This demand will create a tremendous market for software professionals knowledgeable in embedded and cyberphysical systems and their security.

The explosive growth in embedded technology has not been accompanied by substantial educational activities in the area of the security of embedded systems. This project is addressing respective issues by developing a modular security curriculum, with modules accessible directly over the Internet. To meet this objective, a set of dedicated online modules was designed to focus on security of embedded systems. The stated objective of the project was to improve the quality of teaching security for embedded systems in computer science and software engineering. Eight web-based course modules were developed with the following goals:

- studying and learning professional issues in embedded systems security;
- creating lab exercises for practical applications;
- evaluating the project results by professional evaluators; and
- assessing the project's effectiveness and impact in the classroom by the students.

The rest of the paper is organized as follows. First, we present a general description of the module selection procedure and their contents. Next, considerations on pedagogy are outlined, followed by a section on sample module's description in some details. Then, a section on lessons learned in offering the modules in a course is presented, and the paper ends with a conclusion section summarizing the project's results.

Curriculum Essentials and Topics Selection

Security of embedded systems as a subject of an undergraduate course has not been studied that much in the literature, so there are no specific examples to follow. There are some book publications,[2-4] but they address a different type of audience than college students, so by definition are not designed for instruction or teaching related courses. If there are any existing educational publications, they are scarce and hard to find.

With this in mind, designing a related curriculum constitutes a challenge. Specifically, since computer security is such a broad area, involving a multitude of issues, one needs to give some thought to the selection of topics to cover. Even if the application area is narrowed down to just embedded systems, the spectrum of related issues not necessarily shrinks automatically. As an overview of the issues, Table 1 lists chapter titles form the quoted books.[2-4]

Table 1. Chapter titles from books on embedded systems security.

C.H. Gebotys		**D. & M. Kleidermacher**		**T. Stapko**	
Chap.	Title	Chap.	Title	Chap.	Title
1	Where Security Began	1	Introduction to Embedded Systems Security	1	Computer Security Introduction and Review
2	Introduction to Secure Embedded Systems	2	Systems Software Considerations	2	Network Communication Protocols and Built-in Security
3	The Key	3	Secure Embedded Software Development	3	Security Protocols and Algorithms
4	Using Keys	4	Embedded Cryptography	4	The Secure Sockets Layer
5	Elliptic Curve Protocols	5	Data Protection Protocols for Embedded Systems	5	Embedded Security
6	Symmetric Key Protocols Including Ciphers	6	Emerging Applications	6	Wireless
7	Data Integrity and Message Authentication			7	Application Layer and Client/Server Protocols
8	Side Channel Attacks on the Embedded System			8	Choosing and Optimizing Cryptographic Algorithms for Resource-Constrained Systems
9	Countermeasures			9	Hardware-Based Security
10	Reliable Testable Secure Systems			10	Miscellaneous Security Issues and the Future of Embedded Applications Security
11	Summary, Standards, and Ongoing Efforts			11	PIC Case Study
				12	Rabbit Case Study

Reviewing those titles and topics one can see some clusters that can provide certain directions to set up the initial list of modules covering the embedded systems security. First of all, all three books clearly begin with an introduction to the subject, so this would be the primary module of this course: introduction to computer security in general, providing an overview of the issues. Next, each book discusses, in some detail and with different emphasis, the topic of cryptography, with Gebotys' book devoting a significant amount of material to this subject, which is clearly essential to providing security in embedded systems. Therefore, this is a "must" topic for developing another module.

Taking a closer look at the rest of topics (chapter titles) in Table 1 reveals roughly several more specific other areas of interest, which are: methods, network protocols, software, case studies, hardware and some miscellaneous issues. Keeping in mind that this is an undergraduate curriculum, one has to choose topics that are highly relevant but, at the same time, attractive enough to draw students' attention and let them understand the nature of embedded systems security and get the feeling of the associated problems and their solutions, with some minimal practical experience. This is why the second group of topics was selected focused on technology and the third one focused on software.

Thus, topics selected in the first round of the curriculum development included: (1) General educational modules on security, such as "General Introduction to Computer Security", "Introduction to Cryptography" and "Embedded Systems Security"; (2) Security of specific technologies used in embedded systems, which in this edition involve "FPGA Security", "RFID Security" and "SCADA Security"; and (3) Software aspects of embedded systems security, such as "Java Security" and "Threat Modeling." Threat modeling as a subject selected may require some additional justification, because it was not present *per se* in any of the books considered. Nevertheless, when it comes to the exercises and use of tools, essential in computing and software engineering courses, it turns out that threat modeling can be easily implemented with existing free software packages and is very extensively covered in respective books,[5-7] which significantly facilitates the knowledge acquisition process.

The next two sections discuss the approach to pedagogy and using one specific module to present a module structure common to the entire curriculum.

Considerations on Pedagogy

When it comes to pedagogy, the fundamental issue in teaching any subject is to ensure that the students will learn respective material. In engineering and computing disciplines, the reasonable guidelines come from ABET, which can be loosely summarized in the following four bullet points involving all essential elements:

- Learning Outcomes describe learner performance that is expected as a result of learning sometimes called a "competency". It is a discipline-specific major skill, knowledge, or attitude that a student will need to perform a task accurately. It is what students will be able to do as the result of a given learning experience. Competencies are typically broken down into more specific learning objectives.

- Learning Objectives define skills or knowledge that a student will acquire as a step toward an outcome. Objectives are stated in a manner that is clear and measurable providing cues for the development of learning activities.
- Learning Activities are methods that will help students to master specific learning outcomes. Learning activities will guide students through the learning of a competency using structured content presentation and practice.
- Assessment is a process used to provide feedback to both the learner and the teacher about the progress toward understanding intended outcomes. It can be used to adjust teaching and learning in order to maximize learner achievement.

In this view, each module was designed to include respective components, mapped from the above mentioned list, with one exception. Learning Outcomes were considered from the beginning as a meta-goal, which – although critically important – would not constitute itself a part of a module, because Outcomes are determined not at the course level, considered an implementation level, but at the higher level of course design. Thus, each module included the following three components: (1) Objectives; (2) Student Activities (based on Suggested Readings and Hands-on Exercise); and (3) Assessment. All this was enhanced by an independent component called Introduction, whose purpose is to overview the subject and define major concepts to let the student have some background before starting the required reading.

Eight modules were developed that way in this project and have been tested in undergraduate courses on embedded systems and computer networks at one academic institution, Florida Gulf Coast University. They are currently being revised for testing at other universities across the nation. They are all available on the Internet[8] and are currently being expanded to cover aspects specific to cyberphysical systems security.

Sample Module Description – Java Security

In the objectives part of each module, the specific list of objectives is presented, defining general focus of the module. In case of the Java Security module, the objectives state that the student will learn:

- what vulnerabilities exist, exposing a Java program to attacks from an adversary
- what principles are involved in ensuring security of Java programs
- what solutions may developers apply to protect their Java software.

Before assigning any student activities, an Introduction gives the general background on the subject, defining a fundamental question related to the security issue specific to this module. In case of a Java Security module, this part focuses on two aspects: message integrity and message authentication. An excerpt from the Introduction is presented below.

> *Java technology has been a popular tool used to create many stand-alone, enterprise-class and embedded applications to meet the need of its customers. However, there is always a threat from hackers who attempt to invade and steal the important information through any means of the data communication. The main problem when dealing with secure Java programs is securing data transfer between machines on a*

local network or on the Internet. This problem can result in damages to a company or an individual when faced with a security attack. In today's Internet landscape, potential harmful hacking attempts happen all the time, and at times, by random or organized groups. Consequently, there are a variety of problems when it comes to writing secure Java applications for embedded systems.

Since the bulk of Java applications involve communication, one of the principal questions regarding security of Java programs can be stated as follows:

> "How to secure the Java applications so that message sent between Java programs cannot be compromised?"

A part of the answer is that two parameters: message integrity and message authentication are important to make sure no one sees or changes a message sent or received by a program. The integrity of a message is an important step used to check that sent data have not changed when sent to other machines or processes. Simply confirming the message received is accurate is not enough to help prevent issues in programming security, like eavesdropping or message injection. Sometimes simple message authentication procedures are not sufficient and hackers or those who would abuse an application find ways around authentication, such as changing the message digest itself the received message is being compared against. When this is the case, more advanced message authentication tools must be put in place. This could be considered a combination of message encryption and authentication.

For the Student Activities component of each module, there are two basic required parts whose completion assures that the respective knowledge is acquired: (a) Required Reading and (b) Hands-on Exercise. In case of Java Security module, the Required Reading involved the essential mandatory reading from the source "Java Security Overview" white paper,[9] and an optional reading material from a selection of Java security books, determined primarily by their availability in the school library.[10-12]

The reading was accompanied by a specially developed Hands-on Exercise, which involved following the actual code to replicate the security violation incident and curing specific Java vulnerability by providing respective security measures. This example showed how to implement Message Integrity and Message Confidentiality in Java. The sample test applications were written using the Java platform and various extensions, in this particular case those included in JDK 1.4 base: Java Cryptography Extension (JCE), CertPath API, and Java Secure Sockets Extension (JSSE). The sample output of the result of this exercise is shown in Figure 1.

The Assessment component of each module involves three essential elements: (a) Contribution to a Discussion Forum how well the reading has been understood; (b) Taking a Quiz related to the module; and (c) Running a Hands-on Exercise. The weight of each of these components varies, depending on the actual module, since some modules focus more on computational models (for example, Cryptography module, where responding to quiz questions is more appropriate), some focus on understanding the principles (such as, Embedded Systems Security module, in which case expressing views on the Discussion Forum is more appropriate),

and some modules, such as Java Security, focus more on algorithms, in which case following the Hands-on Exercise and actual programming is more important.

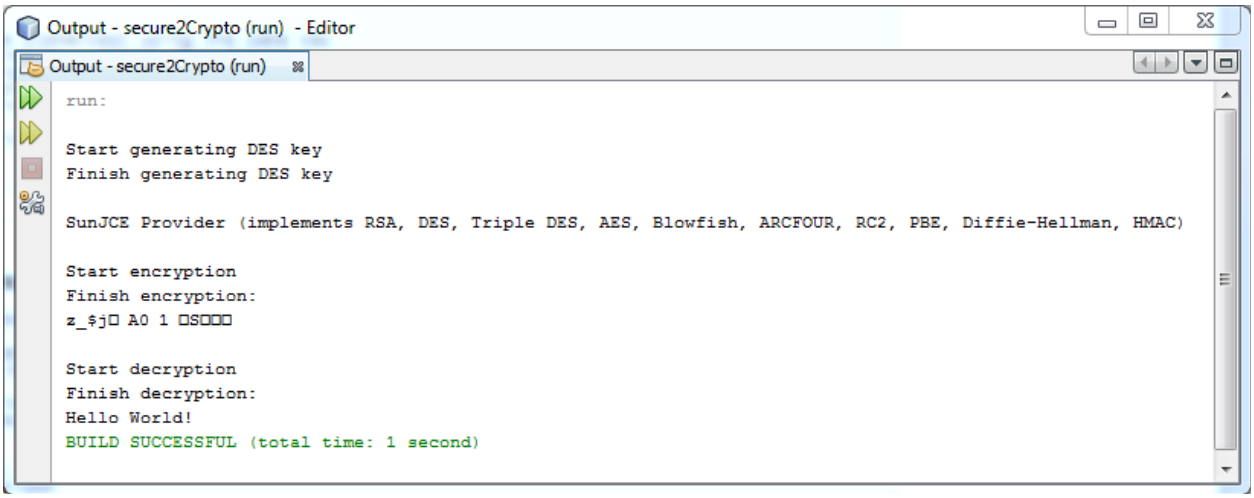

Figure 1. Sample encrypted and decrypted output transmitted with Java exercise.

Overall, even though the general structure of all modules is unified, peculiarities of specific topics can change respective modules' emphasis significantly, leading to some variations in both learning objectives and accomplished results. This is the case, for example, with FPGA Security, due to a very different background (prerequisites) needed to understand the subject matter.

Lessons Learned

All eight modules developed were offered in 2013 in two software engineering courses on an experimental basis, (a) Embedded Systems Programming, and (b) Software Project in Computer Networks (only modules related to networking), to collect feedback and realize the needs for improvement. There have been multiple positive results of modules development and their offering, such as: (a) compacting the knowledge on specific security subjects, usually spread around and hard to identify and extract; (b) relating the general and theoretical knowledge to practice, done here with hands-on exercises; (c) building a model of a security learning module, which, even if imperfect in this edition, can serve as a standard template for further refinement.

However, from the perspective of curriculum improvement the positive results are not that much interesting. Rather, any relevant information on what deficiencies have been discovered and what errors have been made would be of interest to determine how to correct respective issues. In this regard, several important observations have been made.

First, it turned out that the Learning Outcomes component should be made more explicit, rather than somehow concealed in the module content but not identified specifically. The Objectives section of each module identifies the chunks of knowledge related to specific module's material. However, they are not well formulated or expressed in the "learning objective" fashion, because of missing the usual action words "students will be able to ..." that specify what students will be able to do as the result of a given learning experience. The use of a more nebulous phrase "student will learn" makes it more difficult to measure the outcomes and

thus cannot really serve as an objective. In relation to that, since the Introduction section of each module includes a well-defined question, stressing the main topics of the module, it would be an improvement to rephrase the question in the form of the learning outcome, i.e., major skill, knowledge, attitude related to the module (from which learning objectives are extracted).

Next, the sections on Student Activities specify only the assigned reading, leaving the actual activities to the Hands-on Exercise section. It turns out that for more involving activities it may not be quite clear for the student what needs to be accomplished and how to get feedback whether or not the activity was performed to the instructor's satisfaction. Providing some kind of rubrics tied with well-defined learning objectives would likely improve educational aspects of the modules. In addition, Hands-on Exercise of each module presents typically some voluntary student activity pointing to an external material and allowing students their own exploration. A more clear statement on what the student is supposed to do would definitely help in the learning process.

The References section of each module provided a list of readings, but were not strictly enough connected with the module content. This made the students wonder what exactly and how much was expected from them regarding participation in the Discussion Forum. The Assessment section of each module was typically limited to participation in a Discussion Forum and taking of an online quiz consisting of few questions about the material assigned for reading. It was the perception of the students that what would benefit the course is adding some specifics about the method of assessment, such as a process, rubric, evaluation logistics, and especially relating it to the Learning Outcomes.

Conclusion

With the growing complexity of real-time, embedded data acquisition and control applications, software engineers and educators are facing new challenges in securing the development and operation of such systems. The problem is further exacerbated by the widespread use of Internet in connecting embedded devices, which generates additional security issues. This project was aimed to address related problems by developing and testing the skeleton of a respective undergraduate curriculum. Eight modules were developed covering related concepts for a wide range of embedded systems security topics relevant for computer, software, and system engineers entering the modern workforce. They are all available from the project website.[8]

While the first, experimental, offering of these modules in two undergraduate courses was successful, several weaknesses of the modules were identified, especially in their instructional design. The presented modules constitute a good starting point to provide valuable instructional material allowing students to explore the issues of security for embedded systems, but the ways students are tested on the knowledge acquisition need further improvement. As far as the presentation of the material may spark student curiosity and encourage their further exploration of the subject matter, it is questionable if the weaker students, requiring more hand-holding, would use the course material to their advantage.

The clear deficiency of the course material turned out to be lack of well-defined learning outcomes. It was not very obvious what students were supposed to gain as the result of a module

completion. Statements “students will learn about” were difficult to directly relate to assessment. The Activities sections, especially the readings, might have had stronger association to the online testing, which would help evaluate whether or not the assigned activity was accomplished with appropriate understanding. In this view, converting voluntary Hands-on Exercises to more specific activities resulting in better defined assessment procedures would lead to improved practicing of the introduced concepts.

In summary, the entire project turned out to allow students explore this important and typically not well addressed facet of computing education. The scope and technical level of information seemed to be adequate, providing students with enough material to get familiar with the concepts. However, various aspects of pedagogy could be improved, which is the objective targeted in the refinement of the modules for use in the next editions of respective courses and prospective adoption at other universities.

Acknowledgements

This project has been funded in part by the National Science Foundation under Award Number 1129437. Thanks go to Florida Gulf Coast University’s Computer Science and Software Engineering students, Nicholas Alteen, James Carroll, Garret Colas, Vincent Giannone, Anthony Hadding, Christian Paulino and Tyler Thomas, for their contributions to the development of respective modules. Hardware and software donations of respective development platforms from Xilinx and Data Flow Systems are gratefully acknowledged.

References

1. Maybury M.Y., “Air Force Cyber Vision 2025,” Invited Talk, CSIIRW-8, 8th Cyber Security and Information Intelligence Research Workshop, Oak Ridge, Tenn., January 8-10, 2013.
2. Kleidermacher D., M. Kleidermacher, Embedded Systems Security, Elsevier/Newnes, Amsterdam, 2012.
3. Gebotys C.H., Security in Embedded Devices, Springer-Verlag, New York, 2010.
4. Stapko T., Practical Embedded Security, Elsevier/Newnes, Amsterdam, 2008.
5. Howard M., S. Lipner, Writing Secure Code. 2nd Edition, Microsoft Press, Redmond, Wash., 2003.
6. Swiderski F., W. Snyder, Threat Modeling, Microsoft Press, Redmond, Wash., 2004.
7. Howard M., S. Lipner, The Security Development Lifecycle, Microsoft Press, Redmond, Wash., 2006.
8. Florida Gulf Coast University. Software Engineering Program. CEN 3213 Embedded Systems Programming. Topics on Security. Ft. Myers, Florida, December 2013. URL: http://satnet.fgcu.edu/CEN3213/
9. Sun Microsystems. Java Security Overview. White Paper. April 2005. URL: http://www.oracle.com/technetwork/java/js-white-paper-149932.pdf
10. Garms J., D. Siomerfield, Professional Java Security, Wrox Press, Birmingham, UK, 2001
11. Gong L., G. Ellison, M. Dageforde, Inside Java 2 Platform Security: Architecture, API Design, and Implementation. Addison-Wesley, Reading, Mass., 2003.
12. Debbabi M. et al., Embedded Java Security: Security for Mobile Devices. Springer, London, 2007.