Curriculum for Modelling Security: Experiences and Lessons Learned

Haralambos Mouratidis

Innovative Informatics Group, School of Computing and Technology,
University of East London, 4-6 University Way, London, E16 2RD, England
haris@uel.ac.uk

Abstract. The need to develop secure software systems is well recognized by academics and industrialists alike. Current software systems contain sensitive information and therefore it is important that considerable efforts are made to secure such information. To improve the security of software systems, recent research has identified that security analysis should be integrated into software engineering techniques and security should be considered from the early stages of the software systems development process. Although, researchers have focused their efforts towards this direction, recent studies have identified that educational curriculum is not properly addressing this issue. In this experience paper we present the experiences and lessons learned from developing and running a module in Secure Software Systems Engineering at MSc Level.

Keywords: Modelling Security, Curriculum development, lessons learned

1 Introduction

There is general consensus that security of modern software systems is an important issue that needs the attention of everyone involved, either in a research or industrial related role. The last few years, research [1] [2] has argued the need to consider security from the early stages and throughout the software systems development process. Towards this direction, a number of research works [3] [4] have been presented in the literature, and a number of focused events, such as the ModSec 2008 workshop, have been organized. Although some conclusions derived from the published works and the events recognize the need to educate software engineers in considering security through the development process, little effort has been devoted by educational institutions (universities, training academies etc) in developing curriculum to fulfill this need [5].

An increasing number of universities worldwide provide a wide range of security related programmes and modules to tackle various technical aspects of security, such as digital forensics and network security. On the other hand, a wide range of software engineering programmes and modules also exist. However, most of the times, security and software engineering are considered as two separate themes. Although some efforts are made to introduce security related aspects to software engineering modules and programmes, and vice versa; these are individual efforts and they usually focus towards the latest stages of the development stage, such as implementation.

2 The Secure Software Systems Engineering Module

The Secure Software Systems Engineering (SSSE) module is a Masters Level module taught at the University of East London as part of the Internet Systems Engineering MSc programme. The module team consists of three full time academics and, since the module is part of the Internet Systems Engineering programme, the module is focused on the application domain of Internet based software systems.

The module runs over one semester and it has three main learning objectives:

1. To provide students with conceptual knowledge in the analysis, and design of secure software systems;
2. To expose students to the current state of the art in the area;
3. To provide students with practical experience on analysing and designing secure software systems using structured security-aware software engineering methodologies.

The module is assessed by a combination of practical coursework and examinations; and it is delivered via 24 hours of lectures and 36 hours of tutorial and practical sessions. During these sessions, the module covers a wide range of relevant and introductory topics such as computer security and protocols, web security and ethical, legal and professional issues. However, the main topic of the module is on security modelling. This topic includes two main sub-topics. The first one introduces students to the most recent research findings related to the secure software systems engineering area; emphasizing literature on proposed methods, methodologies and modeling languages for modeling security during the development stages of software systems. As part of this, modeling methods such as misuse cases [6] and abuse cases [7]; modeling languages, such as UMLSec [8] and secureUML[9]; and methodologies such as KAOS [10] and Secure Tropos[11] are introduced to the students. The second sub-topic provides detailed description and practical experience of one of the above; the Secure Tropos methodology. The choice for adapting this methodology lies in the wide coverage of the methodology as well as the expertise of the module team on that specific methodology.

3 Secure Tropos

The Secure Tropos software engineering methodology has been presented widely in the literature [11] [12] [13] and it is out of the scope of this paper to replicate this information. The rest of the section aims to provide a brief introduction to the parts of the methodology presented during the module to enable readers to get an idea of what is covered.

During the module, students are introduces to the early and late requirements stages [11] of the methodology as well as the architectural design [11]. The Detailed Design stage and any implementation related stages are not covered. For each of the stages covered, students are introduced to the set of models and the part of the modeling language relevant to that stage. For instance, for the early requirements stage of the methodology, the module team introduces the security enhanced actor [11] and security enhanced goal [11] models along with the corresponding parts of the methodology's meta-model. The students are also introduced to the early requirements stage security-aware process of the methodology. During the tutorial sessions, any questions the students might have, relevant to the theory presented in the lectures, are answered and practical advice is provided on how to create the models. Lessons learned from the application of the methodology to various case studies are also presented. During the practical sessions, the students attempt to use the methodology for the analysis and design of a software system based on a specific case study. It is worth mentioning that since the Secure Tropos methodology is not supported by its own tool, the OME [14] tool is used. Since the tool is not tailored made for the Secure Tropos methodology, the module team has included a session to explain how the tool can be used and how the concepts and models specific to the Secure Tropos methodology can be modeled and developed respectively.

4 Lessons Learned

The discussion in this section is focused on a set of areas that the module team considers important following analysis of the module and feedback from the cohort that studied the module during the second semester of the 2007 – 2008 academic year. Student feedback was obtained through a number of informal sessions/discussions that took place between the module team and the students as well as a number of more formal feedback forms. There were around 35 students registered for the module, coming from a wide range of cultural and educational backgrounds, such as students from Europe, Africa, the Americas and Asia. The students' knowledge and experience in software engineering and/or security engineering was variable ranging from those with previous theoretical and/or practical experience to both areas to those with only very basic theoretical experience in just one of the areas.

4.1 Student perception of the module

Our experience indicates that students' perception of the module changed as the module progressed. Initially, the students were not sure what to expect since none of them had previous experience of a module that integrates security modeling into software engineering practices. However, as the module progressed, all (or almost all) of the students understood the issues involved with modeling security, the challenges faced by the community and the need to have such a module as part of a degree. In addition, we believe there is an association between the perception of the students and their previous studies. It seems that students who came from a software engineering degree/background understood faster the need to integrate security modeling into software engineering practices than those with a stronger security engineering background. For the latter category their perception of security, as a technical only issue, which only requires a set of security mechanisms, was quite strong; and they could not, initially, understand the reason for modeling security from the early stages of the development process.

4.2 Student effort on the module

The student effort on the module increased as their understanding of the need to integrate security modeling into software engineering practices increased. We also found that student effort is associated to the views of students related to their potential careers. Students who became interested in purchasing a career in an area related to secure software engineering were increasingly investing more effort on the module. From student feedback, we identified that a possible way to interest students in pursuing a career related to this area is to present and discuss practical examples of modeling security during software systems development, emphasize studies that demonstrate the cost effectiveness of considering security from the early stages of the development process, and also point out potential post offerings and research and development activity, related to the area, on large companies. Discussing research findings and progress of the research community also made some students interested in pursuing a research career in that area.

4.3 Theory covered

Identifying the right (in terms of quality and quantity) theory to include in the syllabus of a module for modeling security is currently a challenging task. This is mainly because there are no standard or well established approaches and most of the work is still under research development. Therefore, the module was research-driven. The module team tried to covered as much as possible theory related to the area and coming from the research community such as modeling languages, methods, methodologies, security patterns, challenges, and so on. A large number of students, in their feedback forms, indicated that they would prefer to have a more focused syllabus. For example, instead of briefly looking at 3-4 security requirements engineering methods (such as abuse cases, misuse cases, argumentation, security patterns and so on), they would prefer to focus and study, in more depth, 1-2 of them. They stated that this would give them a better understanding of these methods, and therefore feel more comfortable to use them. They felt the pattern followed with the methodologies, where the module focused on the Secure Tropos, should have been followed for the rest of the theory.

4.4 Teaching Resources

Although there is a large number of security engineering and software engineering textbooks, there is lack of appropriate teaching resources that are focused on modeling security. Nevertheless, there is wide representation of research articles in the literature. However, most of these articles quite often make assumptions regarding the knowledge of the readers. Although such assumptions might be valid for the readers coming from the research community, it might not always be the case for students studying the subject. Moreover, there is very limited number of edited books and proceedings with collections of relevant papers. But even those few efforts, have been mostly made with the research community in mind and without considering students studying the area. It is possible that this situation occurs since the area of research is relatively new and most of the proposed approaches are still maturing. This leads to a continuous state of improvement that might prohibit the developers of the various works to document a stable version of their work. It is however an issue that needs to be looked at and appropriate teaching material needs to be developed to assist the development of modules and programmes in the area of secure software systems engineering.

4.5 Assessment

The module was assessed by a combination of coursework and examinations. The coursework was group coursework and was based on a case study that the students were given. The aim was to analyse and design a secure system to meet the case study's requirements. The examination was mainly focused on theoretical questions. The module team felt that it is difficult to assess all the elements of the module using just a group case study coursework and one exam. Although the exam enables the module team to cover all the related theory, it is very difficult to consider all the relevant practical issues just by using one "in house developed" case study. The module team feels that cooperation with industry can greatly assist in defining cases studies that are realistic and cover a large range of practical issues needed.

4.6 Tools

Although some of the approaches presented in the literature have some kind of tool support, most of these tools are prototypes, which in some cases demonstrate some instability. It is however, very important to develop appropriate tools that will be used during classes to enhance the practical understanding that students have of the various approaches. Without such tools, the students fail to understand the usability of an approach and they are usually negative in exploring a method /methodology that does not have even the basic tool support. Student feedback that we received about the need for tools was also focused on the Secure Tropos methodology. Students found it difficult to develop some of the Secure Tropos models, mainly due to the incompatibility of the OME tool to support the development of these models. Therefore, students had to spend a considerable amount of time in creating models, which could have been created much faster and more consistent with the aid of a Secure Tropos tool.

4.7 Exemplars

One of the challenges is to compare the various approaches presented in the literature in an effective and easy for the students to understand way. Usually, in the literature, the usability of an approach (being a method, modeling language, methodology) and its advantages/limitations are described with the aid of case studies and discussions in the various research papers that describe the work. However, sometimes the case studies used and the discussions presented are tailored to emphasise the key characteristics of the approach, and often focus on specific problems. It is important, therefore, to define a suitable example problem (in software engineering community the term exemplar is widely used when referring to an example problem), which will provide a means of comparing various approaches and present such comparisons in a more student friendly way. But apart from the educational value, the definition of an exemplar will also emphasize the problems faced by the community and it will serve as focal point for discussion and exchange of research ideas and results. In choosing an exemplar, various criteria should be considered. For instance, the exemplar should be broad enough to cover all the possible issues, technical or social, that are associated with the development of secure software systems. Moreover, it should be generic enough as well as rich and complex enough to test the limits of any proposed approach.

5 Conclusions

This paper argues for the need to develop curriculum to support education in the area of secure software systems engineering, and it describes lessons learned from teaching a module on secure software systems engineering at MSc level. A number of important issues are discussed in general areas such as student understanding and effort, teaching resources, tools, and assessment. As future work, the module team will be revising some of the syllabus covered in the module to reflect the student feedback and work will commence on developing a tool to support the secure Tropos methodology.

Acknowledgments. The author would like to thank the module team and the students of “SDM032: Secure Software Systems Engineering” module that took place during the second semester of the 2007-2008 academic year at the School of Computing and Technology at the University of East London.

References

1. P. Devanbu, and S. Stubblebine (2000), Software Engineering for Security: a Roadmap, International Conference on Software Engineering, Proceedings of the conference of The future of Software Engineering, Limerick, Ireland.

2. T. Tryfonas, E. Kiountouzis (2001), A. Poulymenakou. “Embedding security practices in contemporary information systems development approaches”, Information Management & Computer Security, Vol 9 Issue 4, pp 183-197.
3. H. Mouratidis and P. Giorgini (eds) (2006) “Integrating Security and Software Engineering: Advances and Future Vision” Idea group, IGI Publishing Group.
4. A. van Lamsweerde (2004), “Elaborating Security Requirements by Construction of Intentional Anti-models”, In Proceedings of the 26th International Conference on Software Engineering (ICSE’04), IEEE Computer Society 148–157
5. B. Whyte, “The teaching of security issues to computing undergraduates in England: a cause for concern”, white paper, http://www.ktn.qinetiqtim.net/content/files/groups/securesoft/SSDSIG_teachingSecurityConcern.pdf
6. G. Sindre, A. L. Opdahl (2005), “Eliciting Security Requirements with Misuse Cases", Requirements Engineering, 10(1), pp. 34-44
7. J. McDermott, C. Fox (1999), “Using Abuse Case Models for Security Requirements Analysis”, In Proceedings of the 15th Annual Computer Security Applications Conference (ACSAC’99)
8. J. Jurjens (2002), “UMLsec: Extending UML for Secure Systems Development”, In Proceedings of the 5th International Conference on the Unified Modelling Language (UML’02). pp. 412–425
9. T. Lodderstedt, D.A. Basin, J. Doser (2002), “SecureUML: A UML-based Modeling Language for Model-driven Security”, In Proceedings of the 5th International Conference on the Unified Modelling Language (UML’02), pp. 426–441
10. A. van Lamsweerde, E. Letier (2000), “Handling Obstacles in Goal-oriented Requirements Engineering”, Transactions on Software Engineering 26(10) pp. 978–1005
11. H. Mouratidis and P. Giorgini (2007) “Secure Tropos: A Security-Oriented Extension of the Tropos methodology” International Journal of Software Engineering and Knowledge Engineering (IJSEKE) 17(2) pp. 285-309, World Scientific.
12. H. Mouratidis, P. Giorgini, and G. Manson (2005) “When Security meets Software Engineering: A case of modelling secure information systems” Information Systems 30(8) pp. 609-629, Elsevier.
13. H. Mouratidis, P. Giorgini (2007) “Security Attack Testing (SAT)-Testing the Security of Information Systems at Design Time” Information Systems 32(8) pp.1166-1183, Elsevier.
14. http://www.cs.toronto.edu/km/ome/