Assurance Cases as a Didactic Tool for Information Security

Roberto Gallo[1,2(✉)] and Ricardo Dahab[1]

[1] University of Campinas, Campinas, SP, Brazil
{gallo,rdahab}@ic.unicamp.br
[2] KRYPTUS Information Security, Campinas, SP, Brazil
gallo@kryptus.com

Abstract. Secure systems are fiercely difficult to obtain - technical, procedural, human, and managerial aspects must be contemplated in a deep, yet holistic approach, which is a complex task even for experienced information security practitioners. Emerging information security "Assurance Cases" methodologies, such as the military NATO AEP-67, promise (time) effective practices for obtaining secure systems, making it a more reproducible process. In this paper we are the first to report the effectiveness of the Assurance Case methodology as a framework for teaching information security to both individuals and teams.

1 Introduction

In spite of over 30 years of research, new information security issues of every nature emerge in a growing rate. Indeed, achieving a secure system is arguably one of the most difficult tasks practitioners may face in their professional lives. Having a secure system demands a mix of procedural, technological, and scientific actions and capabilities that few teams have and even fewer professionals master.

Because of mainstream educational practice limitations, forming professionals that can handle both the comprehensiveness and depth necessary for success in information security is a challenge. These reasons are further explored ahead in this paper.

In this paper we report how Assurance Cases were successfully employed as the technical backbone of a course in secure system conception and implementation, as a means to achieve a holistic approach in teaching information security to both individuals and teams.

This paper is organized as follows: Sect. 2 presents related work, while Sect. 3 introduces the pedagogic model. In Sect. 4 the Information Assurance Case methodology is shortly introduced. The pedagogic model and the chosen assurance methodology form the course syllabus, presented in Sect. 5. The class experience and the experiment evaluation are reported in Sect. 6. Section 7 concludes and presents future work.

R. Gallo—Partially supported by Intel and Samsung research grants.
R. Dahab—Partially supported by CNPq grant 311530/2011-7, FAPESP Thematic Project 2013/25977-7, CAPES grant BEX 7046/14-6, and an Intel research grant.

M. Bishop et al. (Eds.): WISE9, IFIP AICT 453, pp. 15–26, 2015.
DOI: 10.1007/978-3-319-18500-2_2

2 Related Work

Education in information security has been receiving attention for over a decade [15,17]. In general terms, proposals can be classified with respect to four main aspects: duration, scope, integration with other curricula, and the existence of an underlying framework.

Because information assurance is such a broad subject and transversal to most IT-related major degrees, some authors consider that education in this area should start in the freshman year and continue throughout the student's formal education. It may even be possibly offered as a major degree itself [16].

Although ideal, however, this approach requires re-thinking entire curricula, demanding time, effort, faculty mobilization, and other resources. Thus, single courses, or workshop series are the prevailing approaches [13,14,18], as shown in a US survey [19].

When single, self-contained courses are considered, usually there is the need to compromise either in terms of scope or lab practice: as shown in [19], most security courses are in the form of lectures, even though hands-on classes were shown to present very promising results [11,12]. Also, it is important to note that the vast majority of hands-on single class courses are either on attacks, security management or risk assessment topics.

Our goal, however, is broader and builds upon the previous topics: the students, as teams, should become competent on *conceiving, designing, implementing, and evaluating* secure systems. Of course, such an agenda may be hindered by time constraints in a single term course and the inherent complexity of designing new systems.

To overcome these potential problems we adopted a double-edged approach: (a) in order to cope with the challenge of students designing new systems, we adapted a methodology for Electrical Engineering teaching based on product design [9,10], which provided a pedagogical framework for "students developing products"; and (b) in order to cope with time constraints, we took advantage of the Assurance Case Methodology's ability to factor work among students (see Sect. 4).

3 Pedagogic Model

Secure system conception, design and implementation is a complex task that requires creativity and deep and wide knowledge of theoretical and practical aspects. Thus, a traditional, purely narrative class, in the lines of what Freire called "The Banking Concept of Education" [1], is not the most appropriate for teaching these subjects. Although some concepts can and need to be explained to students, our hypothesis is that real knowledge on information security is better internalized by means of experiencing.

For that reason, we employed on our course the concepts of Jean Piaget's constructivism [2] and, more extensively, the theory of Experiential Learning by David Kolb [3] - where knowledge is gained by the appropriation and transformation of the students' experience. And because information security problems are

seldom well structured and transversal to a broad range of areas, we were also inspired by Ivan Monsão's learn-by-doing teaching methodology [9,10], which precognizes that students must be given small yet real problems to work with, in a "close coaching" methodology, so that they may gain hands-on know-how (see Donald Schön [4] and John Dewey works [5]).

4 Assurance Cases - AC

We quote from NATO's Allied Engineering Publication #67 (AEP-67) [8]:

> System assurance is the justified confidence that the system functions as intended and is free of exploitable vulnerabilities, either intentionally or unintentionally designed or inserted as part of the system at any time during the life cycle. This ideal of no exploitable vulnerabilities is usually unachievable in practice, so *programmes*[1] must perform risk management to reduce the probability and impact of vulnerabilities to acceptable levels.
>
> The *Assurance Case* is the enabling mechanism to show that the system will meet its prioritized requirements[...] *It is a means to identify all the assurance claims, and from those claims (formally) trace through to their supporting arguments, and from those arguments to the supporting evidence.*

A key feature of ACs in general is that they support both quantitative and qualitative formal analysis of evaluation criteria, and then combines arguments in a logically structured way. ACs can be represented in different forms, depending on the objectives: graphs (readability), formal language (easy of processing), semi-formal (easy of writing, see Fig. 1). Assurance levels on claims can be presented as probabilities (calculated by logical-probabilistic methods) or simply by labels from risk analysis. We chose AEP-67 as our AC framework as it is both well-documented and a published standard.

A positive yet easy to overlook benefit of using assurance cases is the gained ability to factor both analytical and implementation work on a per component, per requisite (claim), per technology, or per life-cycle fashion, greatly reducing the need for "super-professionals" (with wide and deep knowledge). This is a key enabler of ACs as a methodological tool for education as we can focus on the team.

5 Course Syllabus

Once defined the technical and the educational methods we developed a syllabus where the (bold) objective was to teach students most security aspects of the conception, design and implementation of critical secure systems in a single-term

[1] A set of related measures or activities with a particular long-term aim: e.g. the British nuclear power programme.

- **CLAIM A**: "The software implementation abides to its specifications", with **"medium assurance"**
  - "AND" **SUB-CLAIM-1**: "The software binary correctly corresponds to the source code", with **"high assurance"**
    - **CONTEXT-1.1**: "All source code is interpreted as ISO/IEC 9899:1999 standard";
    - **ARGUMENT-1.1**: "The source code is compiled with a compiler that correctly translates the source code to binaries" with **"high assurance"**
      - **EVIDENCE-1.1**: "The used compiler is CompCert, which is formally verified"
      - **CRITERION**: "Compiler with formal verification" for "high assurance"
  - "AND" **SUB-CLAIM-2**: "The source code abides to its specifications", with **"high assurance"**
    - ...
  - ...
- **CLAIM B**: ...

Fig. 1. Text excerpt from an assurance case. Claim A is mid-level in terms of abstraction and can be a sub-claim of a number of higher-level claims, such as "the system provides only messages with origin authentication". Of course, in order to hold, Claim A also depends on a number of other factors (sub-claims), for example, to guarantee that the source code provided to the compiler is indeed the one intended by the programmer. As a result, the assurance level (or the probability of holding) for a claim with sub-claims is calculated by the composition of probabilities. A complete AC, even for small systems, can have hundreds of elements (claims, sub-claims, arguments, evidences, contexts, criteria, assumptions).

course (60 class hours over 4 months), so efficiency was a major concern. The course was offered both at graduate and undergraduate levels to computer and electrical engineer students and professionals. No specific prerequisites were set other than technical English proficiency.

The course started with 15 enrolled students and was organized in three parts: (A) introduction to security sub-areas, (B) project development, (C) attacks to others students' systems.

Grades were calculated from five main indicators: (i) project adherence to the security goals (claims) and functional requisites, (ii) related AC documentation quality and completeness, with special attention to quantitative and qualitative evidences and evaluation criteria; (iii) attack planning, execution and effectiveness; (iv) bonus questions and quests on selected topics (v) and a final written exam.

5.1 Part A - Introduction to Security Disciplines

This one month long part was basically a sequence of traditional lectures with two objectives: (A) to rise the overall security awareness level, and (B) introduce

the Assurance Case Methodology. An important pedagogic objective was to move students away from Burch's [6] "unconscious incompetence" stage.

The program included: (i) security definitions, (ii) psychology and human factors, (iii) laws and standards, (iv) cryptography, (v) defensive programming, (vi) malwares and example attacks, (vii) side-channel attacks and hardware, (iix) discussion on a sample system, (ix) AEP-67 assurance case methodology.

5.2 Part B - Project Development

At the end of Part A, students were instructed to form two groups around projects. They were presented the option of proposing any non-trivial multi-user system of their choice (for which the teacher would establish security goals) or choose between two "messenger" projects, presented in Table 1, with a common general goal: to provide secure message functionality for a limited-size community of users.

Groups were formed and one leader per group was elected as point of contact and sole "trusted" element on their groups – the class instructor announced that he could covertly designate a team member (other than the leader) as a spy.

Table 1. Proposed project details

Title	"Spartan messenger"	"Athenian messenger"
Allowed limitations	Messages: text only, fixed-size, non-formatted, no history, no message delivery, No timing constraints	Text only messages, No timing constraints
Supporting assumptions	COTs semiconductors are free from targeted menaces, One member per team is trusted	Vanilla computing platform is free from targeted menaces (e.g. factory Android phone), One member per team is trusted, Adversary cannot act upon the hardware internals
Extras	Keep message history	Formatted messages, Voice messages
Security requirements (claims to support)	Data confidentiality, Meta-data confidentiality, Data integrity, Origin and data authentication	Data confidentiality, Meta-data confidentiality, Data integrity, Origin and data authentication, Plausible usage deniability, Replay attack protection
Threat model	Adversary has full power and can do anything other than the supporting assumptions, including deploying spies	Adversary has full power and can do anything other than the supporting assumptions, including deploying spies

This lead students to consider inside-threats from day one on their projects, resulting on a very rich security experience.

During two and half months, students were coached weekly (or twice a week upon request) and both projects and assurance cases were reviewed. Also, students formed e-mail groups for all internal project messages and included the instructor to promote further proximity with them. Technical hints were given whenever sticky points were identified either at live classes or on internal discussions on a close coaching but without direct coding or designing being performed by the instructor.

5.3 Part C - Attacks

Once finished, students provided their resulting project in its "product form" to the other team for security and functional evaluation. The sole provided documentation were user manuals and the security goals "claimed" by each group for their solutions.

Prior to the beginning of the 10-day attack phase, the class instructor introduced students on attack planning, attack surface and effort focus concepts so that they could make the most of their time. Also, instructor insisted in experimentation as a fundamental tool for success (and for learning).

At the end of the final phase, students had the opportunity to fix any security issues of their solutions prior to final grade evaluation.

5.4 Course Conclusion

The course was concluded with a single exam with the following motto: "when security is considered, more important than knowing a subject is the conscience of not knowing it". In Burch's terms, to be unconscious of one's own ignorance is much worse than any other learning phase when it comes to security, as even a single missing aspect is often fatal.

In that exam, although overall grades could range from 0 to 10, individual questions' grades ranged from -32 to $+16$: students were free to choose only the questions they fell comfortable with. This exam structure was carefully chosen so that penalization would not be counterproductive in pedagogic terms, while the nature of information security and related tasks were preserved.

6 Results

6.1 Project and Assurance Case Results

Both teams were able to finish their projects without considerable delays. They employed radically different techniques to achieve their goals with success, generating a handful of creative solutions.

A noteworthy and validating aspect was their ability to conclude, from the assurance case, industry best practices that naturally emerged during its compilation, such as (i) the benefits of the reduction of the trusted computing base, in

a minimalistic approach (see Spartan messenger highlights bellow), (ii) the need for trusted designing tools (see compiler sub-claims on AC excerpt from Fig. 1), (iii) the importance of the supply chain (see Sect. 6.3).

Table 2 summarizes key group achievements and project features and highlights some characteristics.

Athenian Messenger

- **Stealth app launcher.** In order to sustain "plausible deniability" the team devised an app launcher scheme that presented no icons on the Android platform – the application was triggered when a user configurable specific phone number was dialed. The only app trace was a new running process at the device management screen;
- **Anti-permission leakage and interposition (Fig. 2).** The Android platform allows for a number of interposition mechanisms so that regular keyboards can be changed and even entire app screens can be precluded, representing a serious threat. The implemented solution was twofold: (i) when launched, an application scanned all device permissions looking for any hazardous interaction and warning the user if any was found; and (ii) the application was packaged with its own input method so that it was not threatened by rogue keyboards;
- **Dummy server.** Although a server was employed, its sole purpose was of message relaying. Even if the server were compromised, all authentication (data, origin) was performed by the devices themselves and no data breach would occur. To make this possible, a QR-code-based approach was implemented as a means for public-key exchange, when users were adding others to their contact lists;
- **Voice messages.** Groups implemented secure voice message functionality.

Spartan Messenger

- **Minimalistic approach.** Because trust on components should be accompanied by proper evidence, the group decided to have maximum control over their platforms: they acquired Raspberry Pi boards (Fig. 3) and stripped down a Linux distribution specifically for this project. All designed applications were minimalistic written in C language and any non essential feature was removed;
- **Private key protection.** Because private key protection was essential for data and origin authentication, private key was protected by multiple mechanisms, including full disk encryption. A good deal of creativity was employed when individual private keys were embedded on per-user binary application.
- **Assurance Case completeness.** The Spartan messenger team produced a very complete assurance case for their solution. Not only full lifecycle stages were considered, but also good security evidences were provided.
- **Resiliency.** The proposed solution did not rely on any servers, making it immune to some forms of DOS attacks. Because no scalability requirements were set, the group chose a broadcast/multicast network architecture allowing the implementation of strong metadata protection.

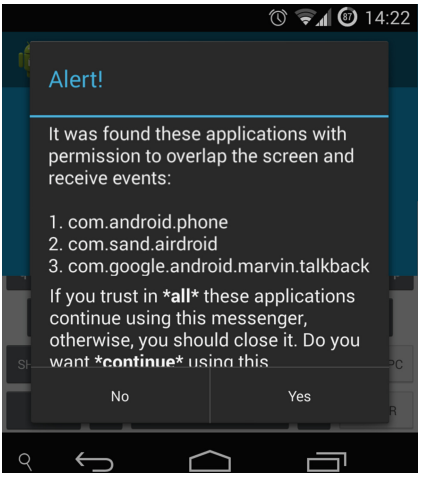

Fig. 2. Figure athenian permission scanner

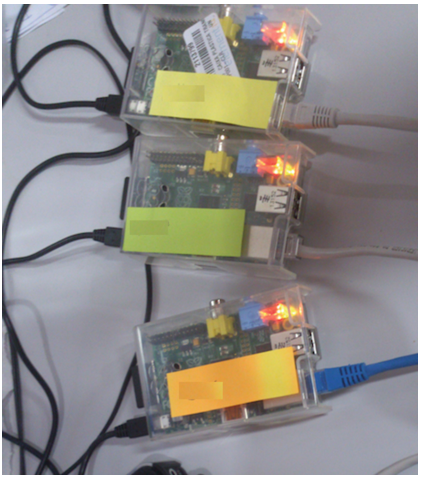

Fig. 3. Figure spartan messenger hardware clients

Table 2. Key project and documentation results

Title	"Spartan messenger"	"Athenian messenger"
Chosen platform	Raspberry Pi + stripped Linux	Android phone
Architectural approach	Minimal trusted computing base	Security in layers
Cryptographic techniques for data in transit	Designed their own protocol using standard primitives. OpenSSL crypto library as core (SSL was not used)	Mixed their own protocol with Onion routing. GPG crypto core
Data at rest and binary protection	No user data at rest. Protected binaries by file system permissions and full disk encryption	SQCipher for database protection, with key derivation from user's PIN
Anti-metadata and side channel protection	Fixed size, fixed time package transmission with broadcast	Fixed size, fixed time package transmission with relay server
Highlights	Assurance case naturally showed strong security dependence over the supply chain. Very complete assurance case, split by lifecycle stage	Android permission scanning prior to application initialization, look for rogue apps. Innovative app launching scheme for plausible deniability

6.2 Attacks

Although only 10 days were given for attacking, teams deeply and consistently explored each others' solutions, giving rise to a number attack trials: 19 for Spartan team attacking Athenian app and 10 for Athenian team attacking Spartan solution.

Athenian Team on Spartan Solution. The attacks performed by the Athenian Messenger Team were distinguished by their systematization and the ability to find weaknesses (but not a violation) on the Spartan Messenger solution claims.

In terms of systematization, The Athenian Messenger Team organized potential attacks depending on: (a) attack surface, (b) active vs passive adversary model, (c) complexity, and (d) execution time. Of course, attack plans evolved while they learned more about the Spartan solution.

They performed analysis on network traffic, hardware I/O interfaces, the Linux image, and the application binaries, among others. They could find a sole marginal weakness - although no security claim was violated: on the Spartan solution, a user sends messages to other users using a command line application, passing message and receiver as arguments (e.g. ./sender receiver_name message). However, because the Spartan Team forgot to disable the BASH history, past messages persisted on the .bash_history file – this is only a moderate problem given that full disk encryption was in place.

The BASH history problem was solved for the final solution version.

Spartan Team on Athenian Solution. The Spartan Messenger Team spent considerable analytic and coding effort when attacking the Athenian Messenger. The first step was to decompile the application package (APK) so that further knowledge of the underlying cryptographic protocol was gained.

Although the protocol correctly addressed confidentiality, and data and origin authentication, it was susceptible to replay attacks. So, the decompiled code was used again, but this time to develop a Athenian malware that was able to perform replay attacks if the adversary could manage to be included as a contact into others' contact list. This was a violation of one of the intended security claims for the Athenian solution.

Interview with the Athenian team showed what Monsão calls "Aladdin Effect" as one of the root causes of the protocol defect: taking autentication mechanisms as black boxes instead of understanding their internals led to a deformed perception of its behaviour. Once students were presented to the precise mechanism functionality, they promptly corrected the protocol.

6.3 Student Perceived Evolution

In individual interviews, students reported that the Assurance Case methodology was an essential tool for systematizing protection mechanisms' coverage on their solutions.

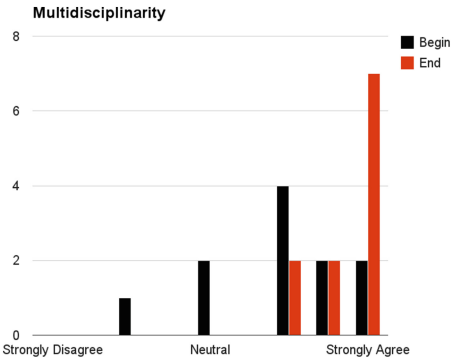

Fig. 4. Information security is a multidisciplinary area

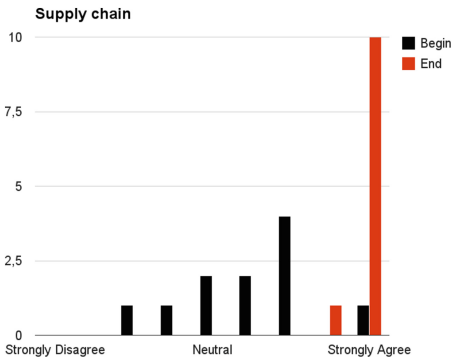

Fig. 5. The supply chain aspects are fundamental to information security

In order to proper capture student perception of their evolution, we conducted an anonymous and optional survey on their perception regarding seven relevant security aspects, comparing the perception they had at the beginning and at the end of the course. We used the Likert scale [7], with "1 for strong disagree", and "9 to strongly agree". Concordance with the following assertions were evaluated:

1. "Information security is a multidisciplinary area";
2. "Psychology [and social] aspects are fundamental to security";
3. "The supply chain aspects are fundamental to information security";
4. "Guaranteed security is a near impossible objective";
5. "Managerial methodologies are fundamental do security";
6. "Solution architecture is fundamental to security";
7. "Secure development methodologies are fundamental to security";

Figures 4 and 5 show the evolution for aspects 1 and 3 respectively, formatted as histograms. Although the number of samples is small, the course's influence is clear. Table 3 shows statistics for all seven assertions for the general group

Table 3. Statistics for assertion submitted to Likert-scale evaluation by students. Cell elements are organized in pairs: the first number is the average and second is the median.

Population	1	2	3	4	5	6	7
All begin	6.8-7	6.7-7	6.0-6	5.2-5	5.9-6	7.8-8	6.8-7
All end	8.5-9	8.5-9	8.9-9	7.5-8	8.5-9	8.6-9	8.4-9
Pro begin	7.6-7	7.6-8	6.2-7	5.6-6	5.4-6	8.2-8	6.0-7
Pro end	8.8-9	8.4-8	9.0-9	7.4-9	8.2-8	8.6-9	8.0-9

and a cut with only the students that work with security in a regular basis (professionals, with 4.8 year experience on average - 5 year median). There is no sensitive difference between the general group and the second group: both groups had seen similar improvements.

7 Conclusion

Although complex, learning how to conceive, design and implement secure systems can be achieved with a proper mix of baseline security awareness, coaching and managerial methodology. In this paper we reported the first (to the best of our knowledge) use of information assurance methodology as a backbone for security teaching. Both self-perception evaluations (with Likert scales) and practical results showed that students were able to internalize hands-on knowledge on the subject. Nevertheless, there is room for improvement - the attack phase was of intense learning, but its duration was reduced: the development phase took considerable time and closer schedule control would allow for smaller delays. Also, although course results are consistent, the sample size in terms of students is small.

Acknowledgement. We would like to thank our students for their dedication and the rich feedback they provided. We also like to thank Ivan Monsão for sharing his experience and feedback related to his pedagogic model.

References

1. Freire, P.: Pedagogia do Oprimido Paz e Terra, Rio de Janeiro (1970)
2. Mussen, P.H. (ed.): Piaget's Theory in Carmichael's Manual of Child Psychology, vol. 1. Wiley, New York
3. Kolb, D.A.: Experiential Learning: Experience as the Source of Learning and Development. Prentice-Hall, New Jersey (1984)
4. Schän, D.A.: The Reflective Practitioner: How Professionals Think In Action, Basic Books (1984). ISBN-10: 0465068782, ISBN-13: 978–0465068784
5. Archambault, R.D. (ed.): John Dewey on Education: Selected Writing. University of Chicago Press, Chicago (1974)
6. Adams, L.: Learning a new skill is easier said than done. http://www.gordontraining.com
7. Likert, R.: A technique for the measurement of attitudes. Arch. Psychol. **22**, 5–55 (1932)
8. North Atlantic Treaty Organization - NATO: AEP-67 ENGINEERING FOR SYSTEM ASSURANCE IN NATO PROGRAMMES, February 2010
9. Monsão, I.: Uma Nova Metodologia de Ensino de Engenharia Elétrica Usando um Laboratório Paradidático, Ph.D. thesis, University of Campinas (2014)
10. Monsão, I.C., Dias, J.A.S., de Jesus F Cerqueira, J., da Costa, A.C.P.L.: A new methodology to teach electrical engineering using product development projects. In: 2012 IEEE Engineering Education: Innovative Practices and Future Trends (AICERA) (2012)

11. Sharma, S.K., Sefchek, J.: Teaching information systems security courses: A hands-on approach. Comput. Secur. J. **26**(4), 290–299 (2007). doi:10.1016/j.cose.2006.11.005. Elsevier
12. Wu, D., Fulmer, J., Johnson, S.: Teaching information security with virtual laboratories. In: Innovative Practices in Teaching Information Sciences and Technology, pp. 179–192. Springer International Publishing, Switzerland (2014). doi:10.1007/978-3-319-03656-4_16
13. Gandhi, R., Jones, C., Mahoney, W.: A freshman level course on information assurance: can it be done? here's how. ACM Inroads, ACM, September 2012. doi:10.1145/2339055.2339072
14. Logan, P.Y., Clarkson, A.: Teaching Students to Hack: Curriculum Issues in Information Security, SIGCSE Bull, ACM, February 2005. doi:10.1145/1047124.1047405
15. Maconachy, W.V., Schou, C.D., Ragsdale, D., Welch, D.: A model for information assurance: An integrated approach. In: Proceedings of the 2001 IEEE, Workshop on Information Assurance and Security. United States Military Academy, West Point, 5–6 June 2001
16. Howles, T., Romanowski, C., Mishra, S., Raj, R.K.: A holistic, modular approach to infuse cybersecurity into undergraduate computing degree programs. In: Annual Symposium On Information Assurance (ASIA), Albany, NY, 7–8 June 2011
17. Crowley, E.D.: Information system security curricula development. In: Proceedings of the 4th Conference on Information Technology Curriculum, CITC4 2003, ACM (2003). doi:10.1145/947121.947178
18. Pattinson, M.: CISA: COBIT: An ideal tool for teaching information security management. Inf. Syst. Control J. **6**, 33–36 (2004)
19. Manson, D.P., Curl, S.S., Torner, J.: A framework for improving information assurance education. Commun. IIMA: **9**(1), Article 6 (2009)