CyBOK Report on Classroom Usage of Case Studies

Nancy R. Mead
nrmcmu@gmail.com

Bastian Tenbergen
bastian.tenbergen@oswego.edu

December 2021

Table of Contents

Abstract ***3***

1 Introduction ***3***

2 Background & Related Work ***4***

2.1 The CyBOK Version 1.1 ..... **4**

2.2 Case Studies and Summative Learning in SE Education **5**

3 Overview of CyBOK Case Studies ***6***

3.1 **Common Case Study Structure** **11**

3.2 Development Process and Quality Criteria **12**

3.3 Mapping to CyBOK Knowledge Areas **12**

4 Preliminary Objective Results from Applying the Case Studies ***14***

4.1 Driver Assistance System Case Study **15**

4.2 Results and Experiences from Applying the Driver Assistance System Case Study **16**

5 Subjective Case Study Classroom Experiences ***18***

5.1 Acme Water (by Shamal Faily) **18**

5.2 GPS Spoofing of UAV (by Carol Woody) **19**

5.3 Organization Risk Management: The Widget Company Case Study (by Carol Woody) **20**

5.4 Penetration Test Case Study (by Bastian Tenbergen) **20**

5.5 Secure Acquisition Case Studies (by Dan Shoemaker and Anne Kohnke) **21**

5.6 SQUARE Case Study (by Nancy Mead) **23**

6 Conclusion and Future Work ***24***

Acknowledgements ***24***

References ***24***

Abstract

One of the central aspects of specialization in modern software engineering is security engineering. With contemporary systems being networked and entrusted with mission-critical functionality, cybersecurity is an essential quality that must be developed into the system from the first moment. This comprises issues such as privacy, authentication, robustness against vulnerabilities, and hardness against external attacks. To do so, software engineering specialists with appreciation for the detailed intricacies of security engineering as well as broad experience are required. The Cybersecurity Body of Knowledge (CyBOK, [1]) has been developed to serve, among other uses, as an instructional reference for educators to prepare the next generation of security engineers in this respect.

While the CyBOK describes the intricacies of security engineering in plentiful detail, it remains up to the instructor to convey this curriculum in a way that fosters understanding and forms experience as well as competencies in the learner. To aid the instructors who use the CyBOK, we have devised a library of 25 case studies that are specifically designed to target CyBOK knowledge areas. The case studies are sufficiently detailed to allow adoption with minimal overhead on the instructor. In this report, we describe the case study mapping to the CyBOK, and classroom results, including both objective results of one exemplary case study, demonstrating improved understanding by students, as well as subjective results of case study usage.

1 Introduction

As the increase and dependence on digitally enabled technology continues to impact almost every area of life, it has created a demand for innovative software-based solutions. However, developing secure software is a multi-faceted activity that can strain a project's budget, design, and overall functionality [2]. The demand for software often pits delivering value at high speed against high quality. In 2020, poor quality software cost organizations $2.08 trillion in the United States alone [3]. The U.S. government tracks software vulnerabilities in their National Vulnerability Database, which is fed by the Common Vulnerabilities and Exposures list. By 2020, more than 18,000 software code vulnerabilities had already been included [4].

In her 2000 paper, Mary Shaw [5] called, among other things, for software engineering education to start at the earliest feasible point during the students' university career and to seek out ways to improve role-specific software engineering education. Now, more than 20 years later, her call has been answered with many software engineering curricula offering broad experiences as well as avenues for specialization, for example, in requirements engineering [6], [7], testing [8], or supply chain risk management [9], [10]. Yet, in today's rapid development environment, security engineering has become a specialization that will only grow in demand [11]. As modern systems are increasingly interconnected and exchange mission-critical, confidential data with one another, they become attractive targets for attackers. Hence, systems must be sufficiently hardened against any type of vulnerability.

Designing such systems requires a substantial amount of security-relevant knowledge, attention to detail, and a considerable level of experience. To help educate the new generation of security engineers, a recent effort lead by the University of Bristol compiled and produced a substantial resource called the "Cybersecurity Body of Knowledge" (CyBOK, [1]). CyBOK 1.1 is structured in five parts and 21 chapters, each of which suggests knowledge areas related to social, organizational, technical, and procedural issues in cybersecurity. CyBOK is intended to serve as a reference curriculum and resource material for instructors to structure cybersecurity education.

Yet, faculty developing new courses on the topic might additionally require suitable resource artifacts to foster summative learning (as opposed to formative learning, e.g., through rote memorization of required reading [12]). Resource artifacts may comprise case studies, homework assignments examples, and assessment options such as exams. These artifacts, while sometimes publicly available, are often

buried in complete sets of course material passed from one instructor to another and are not documented in a consistent or necessarily usable format.

To alleviate this issue, we present a library of ready-to-use case studies in this paper, tailored to select CyBOK knowledge areas. Case studies are derived from and describe real-world examples and resources or rich, fictive contexts. They feature assignment descriptions and application guidelines for the instructors as well as example solutions (if applicable) and/or assessment criteria. Herein, we give a brief overview of the case studies included in our library, their mapping to the CyBOK curriculum, and give an example of their initial application, including results.

This report is structured as follows. Section 2 gives some background on the CyBOK and reviews the related work on case study application in Software Engineering Education. Section 3 overviews our library and associates the case studies with CyBOK learning objectives. In Section 4, we discuss objective results of case study usage in the classroom, Section 5 includes subjective results of other case study usage, and Section 6 concludes this report with an outlook on future work.

2 Background & Related Work

In this section, we briefly introduce the CyBOK. We also discuss the use of case studies in software engineering education.

2.1 The CyBOK Version 1.1

The Cyber Security Body of Knowledge Version 1.1 (CyBOK) is a freely accessible community resource funded by the National Cyber Security Programme in the United Kingdom and published under the Open Government License [1]. CyBOK is an attempt to consolidate cybersecurity as a discipline, which in the past has been fragmented [13]. By contrast, in fields such as software engineering, computer science, or chemistry, there have been collaborations with leading professional societies that have codified key foundational knowledge on which educational programs have been designed and developed (e.g., the Software Engineering Body of Knowledge, SWEBOK, see [14]). Other efforts have established skills, tasks, competencies, risk, and cyber frameworks that exposed many facets to the discipline [15]. A more recent global undertaking with four leading professional societies and a host of academics and practitioners forming a Joint Task Force, resulted in a comprehensive curricular volume to structure the cybersecurity discipline and provide guidance for cybersecurity education [16]. However, among the diverse community of academics, practitioners, and researchers, there has not been progress in reaching a consensus of what is considered the foundational knowledge in cybersecurity [13], [16].

An analysis of the Joint Task Force work along with the ACM Computing Classification System taxonomy, technical certifications, calls for papers, standards, and tables of contents in a variety of textbooks were text-mined using natural language processing and automatic text clustering to group relevant topics and identify the relationships between the topics. Consulting with academics, practitioners, key experts, as well as garnering community feedback, the CyBOK Version 1.0 was developed, and subsequently updated to CyBOK Version 1.1. 21 Knowledge Areas (KAs) form the scope of the CyBOK [1]. The 21 KA are grouped into the following five categories, as shown in Listing 1. Note, that the numbering scheme herein adopts the chapter numbers from CyBOK, and therefore starts at “2”.

I. Human, Organisational & Regulatory Aspects
    2. Risk Management and Governance
    3. Law & Regulation
    4. Human Factors
    5. Privacy & Online Rights
II. Attacks & Defences
    6. Malware & Attack Technologies
    7. Adversarial Behaviour
    8. Security Operations & Incident Management
    9. Forensics
III. Systems Security
    10. Cryptography
    11. Operating Systems & Virtualisation
    12. Formal Methods for Security
    13. Distributed Systems Security
    14. Authentication, Authorisation & Accountability
IV. Software Platform Security
    15. Software Security
    16. Web & Mobile Security
    17. Secure Software Lifecycle
V. Infrastructure Security
    18. Applied Cryptography
    19. Network Security
    20. Hardware Security
    21. Cyber-Physical Systems Security
    22. Physical Layer & Telecommunications

Listing 1 CyBOK Categories, Knowledge Areas, and corresponding Chapter Numbers.

A detailed description of the categories, knowledge areas is available in the CyBOK Version 1.1 companion text [17].

2.2 Case Studies and Summative Learning in SE Education

It is widely accepted in the education literature [18], that strictly relying on formative learning approaches leads to poor theory retention beyond the end of instruction [19]. “Formative” in this sense encompasses lecturing, rote memorization, and high-stakes assessments (e.g., a single exam to determine grades). Instead, summative approaches have been frequently proposed in a variety of disciplines [20], [21], including software engineering [22], and their use in conjunction with formative learning is advocated [12]. “Summative” refers to stimulating knowledge discovery, e.g., by using real stakeholders [7], industry-realistic projects [19], low-stakes assignments [23], games [24], or collaborative teams [25].

Effective instruction of summative approaches, however, relies on a solid theoretic foundation, which is why a combination of both educational styles is required [18], especially in disciplines, in which application and experiences outweigh theory in terms of value for the learner [12], such as software engineering. To achieve this, illustrative non-trivial examples such as vignettes and case studies are an effective tool. The difference between the two is essentially their complexity. Yet, vignettes are usually too brief to provide a proper context and thereby require a thorough investigation of the problem to arrive at a proper solution [26]. An example of a vignette could be: “Paul holds open the door to the clearance level 3 office doors for Jamie, who walks right behind Paul, such that Jamie doesn’t have to swipe her access card. Describe the security-related problem with Paul’s behavior.”

Case studies [27] on the other hand do not have this limitation. Case-based teaching has been part of instructional pedagogy since the late 1880's, primarily starting with law courses and later adopted by both schools of business and medicine from the early 1900's forward [28]. Written case studies can range from a brief outline to illustrate a theoretical point to more elaborate cases, organized and separated into sections with relevant questions and discussion points to integrate theoretical and practical content. What constitutes a case study in education depends largely on the educational goals and pedagogical approach, and of course the instructor. Educational case studies must not be confused with using case studies for empirical evidence in software engineering research (cf. [29]). Case study-based instruction may comprise video materials, curriculum modules, and educational materials for faculty use in software engineering courses [30], ranging from case studies to entire video courses for classroom use. What case studies in this sense have in common is that (a) they provide sufficient context for a problem domain, often involving fictive examples or real-world cases from journalistic or popular scientific sources, and (b) provide task descriptions for students that allows exploring several alternative, equally acceptable solutions rather than one ideal solution.

Application areas of case studies in software engineering include, for example, efforts to increase student motivation for theoretical concepts in requirements engineering [19] or software engineering [31], validation and verification activities [32], and also cyber security engineering [33]. Case studies have a thoroughly positive influence on learning outcomes as concepts are more easily adopted [31], yet occasionally at the expense of an (often unfounded) anxiety over students' final grades [34]. One reason for the popularity of software engineering case studies is that they represented real situations that may be encountered in practice [35]. Such findings have carried over to our experience teaching cybersecurity courses (see, e.g., [36]), where realistic case studies resonate more with students than artificial problems. Yet, the development of a library of case studies specific to a curriculum is a novel approach, in the experience of the authors.

3 Overview of CyBOK Case Studies

Our aim in creating a library of case studies and mapping them to the CyBOK, was to provide educators with relevant and high-quality materials which they can use 'as is' or customize for use in their classrooms. An advisory board consisting of volunteer faculty members and experts in systems, software, and cybersecurity worked to create a collection of robust case studies and citations, thus saving faculty members from having to do the work of researching and structuring the case studies for instructional use or developing their own. Most realistic case studies do not have a single "correct" solution, but where sample solutions exist, these accompany the case studies. Listing 2 itemizes the case studies and the level 2 topic areas that they can be applied to. The version number in the parentheses reflects the CyBOK version for which the respective case study was created, however we believe each case study is suitable for any version.

ACME Water Case Study (since CyBOK 1.0)

This case study has 10 separate exercises that span the following CyBOK topic areas:

Exercise 1: Introduction & Human Error

I. Human, Organisational & Regulatory Aspects

2. Risk Management & Governance.
4. Human Factors

Exercise 2: Risk & Trust

I. Human, Organisational & Regulatory Aspects

2. Risk Management & Governance

IV. Software Platform Security

17. Secure Software Lifecycle

Exercise 3: Personas

I. Human, Organisational & Regulatory Aspects

4. Human Factors

Exercise 4: Requirements

I. Human, Organisational & Regulatory Aspects

4. Human Factors

Exercise 5: User Interfaces

I. Human, Organisational & Regulatory Aspects

4. Human Factors

Exercise 6: Architecture

IV. Software Platform Security

17. Secure Software Lifecycle

Exercise 7: Authentication

I. Human, Organisational & Regulatory Aspects

2. Risk Management & Governance

III. Systems Security

14. Authentication, Authorisation & Accountability

Exercise 8: Authorisation

III. Systems Security

14. Authentication, Authorisation & Accountability

Exercise 9: SEAT & Privacy

I. Human, Organisational & Regulatory Aspects

4. Human Factors

5. Privacy & Online Rights

IV. Software Platform Security

17. Secure Software Lifecycle

Exercise 10: Economics & Entrepreneurship

I. Human, Organisational & Regulatory Aspects

4. Human Factors

Aircraft Service Application Case Study (since CyBOK 1.0)

IV. Software Platform Security

17. Secure Software Lifecycle

Archetypal Users—Personae non Gratae (PnGs) Case Study (since CyBOK 1.0)

I. Human, Organisational & Regulatory Aspects

2. Risk Management and Governance

Deciphering Case Study (since CyBOK 1.1)

II. Attacks & Defences

6. Malware & Attack Technologies

III. Systems Security

10. Cryptography
11. Operating Systems & Virtualisation
13. Formal Methods for Security

V. Infrastructure Security

18. Applied Cryptography

Driver Assistance System Safety & Security Case Study (since CyBOK 1.0)

I. Human, Organisational & Regulatory Aspects

5. Privacy & Online Rights

III. Systems Security

12. Distributed Systems Security

IV. Software Platform Security

15. Software Security
16. Web & Mobile Security

V. Infrastructure Security

20. Hardware Security
21. Cyber-Physical Systems Security

Drone Swarm Case Study (since CyBOK 1.0)

IV. Software Platform Security

17. Secure Software Lifecycle

FAA ERAM Outage Case Study (since CyBOK 1.0)

I. Human, Organisational & Regulatory Aspects

2 Risk Management and Governance
4 Human Factors

IV. Software Platform Security

15. Software Security

GPS Spoofing of UAV Case Study (since CyBOK 1.0)

I. Human, Organisational & Regulatory Aspects

2 Risk Management and Governance

Heartland Payment System Breach Case Study (since CyBOK 1.0)

II. Attacks and Defences

7. Adversarial Behavior
8. Security Operations & Incident Management

III. Systems Security

11. Operating Systems and Virtualization
14. Authentication, Authorisation, & Accountability (AAA)

Mt. Gox Bitcoin Theft Case Study (since CyBOK 1.0)

II. Attacks and Defences

6. Malware & Attack Technologies
7. Adversarial Behavior
8. Security Operations & Incident Management
9. Forensics

III. Systems Security

10. Cryptography
14. Authentication, Authorisation, & Accountability (AAA)

National Cybersecurity Governance Case Study (since CyBOK 1.1)

I. Human, Organisational & Regulatory Aspects

2. Risk Management & Governance
3. Law & Regulation

II. Attacks and Defences

8. Security Operations & Incident Management

National Grid SAP Adoption Case Study (since CyBOK 1.0)

I. Human, Organisational & Regulatory Aspects

2. Risk Management and Governance

IV. Software Platform Security

17. Secure Software Lifecycle

Organization Risk Management: The Widget Company Case Study (since CyBOK 1.0)

I. Human, Organisational & Regulatory Aspects

2. Risk Management and Governance

Penetration Test Case Study (since CyBOK 1.1)

I. Human, Organisational & Regulatory Aspects

2. Risk Management & Governance
5. Privacy & Online Rights

II. Attacks & Defences

6. Malware & Attack Technologies
7. Adversarial Behaviour
8. Security Operations & Incident Management

III. Systems Security

10. Cryptography
11. Operating Systems & Virtualisation
14. Authentication, Authorisation, & Accountability

IV. Software Platform Security

14. Software Security

V. Infrastructure Security

18. Applied Cryptography
22. Physical Layer & Telecommunications

Ransomware Case Study (since CyBOK 1.1)

I. Human, Organisational & Regulatory Aspects

2. Risk Management & Governance
3. Law & Regulation

II. Attacks & Defences

6. Malware & Attack Technologies
7. Adversarial Behaviour
8. Security Operations & Incident Management

Role Based Access Control Case Study (since CyBOK 1.1)

I. Human, Organisational & Regulatory Aspects

5. Privacy & Online Rights

III. Systems Security

15. Authentication, Authorisation, & Accountability

IV. Software Platform Security

15. Web & Mobile Security

V. Infrastructure Security

19. Network Security

Secure Acquisition Case Study 1: Project Initiation (since CyBOK 1.0)

IV. Software Platform Security

17. Secure Software Lifecycle

Secure Acquisition Case Study 2: Acquisition/SCRM Project Risk Analysis (since CyBOK 1.0)

IV. Software Platform Security

17. Secure Software Lifecycle

Secure Acquisition Case Study 3: Adequacy of Acquisition Practice (since CyBOK 1.0)

IV. Software Platform Security

17. Secure Software Lifecycle

Secure Acquisition Case Study 4: Supplier Capability Evaluation (since CyBOK 1.0)

IV. Software Platform Security

17. Secure Software Lifecycle

Secure LAN Case Study (since CyBOK 1.1)

I. Human, Organisational & Regulatory Aspects

2. Risk Management & Governance

III. Systems Security

11. Operating Systems & Virtualisation
12. Distributed Systems Security

IV. Software Platform Security

14. Software Security
15. Web & Mobile Security

V. Infrastructure Security

18. Network Security
22. Physical Layer & Telecommunications

SQUARE Case Study (since CyBOK 1.0)

IV. Software Platform Security

17. Secure Software Lifecycle

Tokeneer ID Station Project Case Study (since CyBOK 1.0)

III. Systems Security

13. Formal Methods for Security

IV. Software Platform Security

17. Secure Software Lifecycle

Using Malware Analysis to Improve Security Requirements Case Study (since CyBOK 1.1)

II. Attacks and Defences

6. Malware & Attack Technologies

IV. Software Platform Security

17. Secure Software Lifecycle

Wireshark Case Study (since CyBOK 1.1)

II. Attacks and Defences

6. Malware & Attack Technologies
9. Forensics

III. Systems Security

12. Distributed Systems Security

V. Infrastructure Security

19. Network Security
22. Physical Layer & Telecommunications

Listing 2 Case studies and their related topic areas

3.1 Common Case Study Structure

Most of the case studies share a common structure to foster quick and easy adoption by the instructor. The subsections that comprise the format of the cases are as follows:

Background. This section provides a brief overview of the real-world and/or fictional example at hand and provides sufficient context to frame the problem space. This section makes references to externally available resources, if applicable, or suggests further reading.

Case Study Overview. This section takes a step back from the subject matter provided in the "Background" section and describes the learning activities to be carried out on the basis of the information given in "Background" to meet CyBOK learning outcomes.

Student Instructions. As the name suggests, this section contains concrete work assignments for students with sufficient detail to understand what is expected but with enough leeway to allow the learner to explore the problem space. This section may be subdivided into multiple tasks or provide partial solutions to get started.

Instructor Notes. This section discusses pedagogical strategies on how to apply the case study. For example, this may entail ways to tailor one case example for group vs. individual project assignments or exam questions, or solution templates.

Example Solution. If one is available, this subsection contains example solution(s), key grading criteria, success factors, or caveats depending on the case study at hand.

References. This section contains references to external resources and/or further reading.

All case studies are freely available and non-commercial usage is permitted, provided respective copyright and attributions are honored. An example case study is provided and discussed in Section 4.

3.2 Development Process and Quality Criteria

The case studies were created based on the respective author's experience and knowledge of the subject matter. Each (team of) authors presented the advisory team with a choice of case study topics, along with a brief overview of the content. Once approved, case studies were independently worked on by each author during the Spring and Fall 2021 semesters. At regular intervals, status updates were reported to the Principal Investigators or advisory team, who would then ensure that the following quality criteria were met. Specifically, each case study was designed and formatted to:

1. Involve real-world examples from journalistic or popular scientific sources or fictional examples sufficient to provide a rich problem space.
2. Provide sufficient context for a problem domain by referencing said sources or providing sufficient explanation.
3. Provide detailed task descriptions that allow exploration of alternative solutions.
4. Provide instructor guidance on how to apply the case study in a given educational setting; and
5. Contain example solution descriptions, common pitfalls to avoid, and/or critical success factors to attain (if applicable, given the case study topic).

Principal Investigators or members of the advisory board validated the submissions against the above criteria, where possible enforced a common structure, and maintained a reporting structure pertaining to the mapping of each case study to CyBOK knowledge areas.

3.3 Mapping to CyBOK Knowledge Areas

One goal of this project was to collect many useful and completed case studies, specific to the software security engineering discipline, and map them to the knowledge areas of the CyBOK [17]. We hoped that a large quantity of case studies would cover the entire CyBOK with multiple case studies, thus providing variety through different examples and assignments for many knowledge areas. The result was 25 different case studies covering all of the CyBOK 1.1 knowledge areas, as shown in Table 1. The version number in the rightmost column reflects the CyBOK version from Listing 2.

Table 1. Mapping of Case Studies to CyBOK Knowledge Areas

Cat.	Knowledge Area	Case Study Mapping	CyBOK version
Human, Organizational and Regulatory Aspects	Risk Management & Governance	ACME Water	1.0
		Archetypal Users – Personae non Gratae	1.0
		FAA ERAM Outage	1.0
		GPS Spoofing of UAV	1.0
		National Cybersecurity Governance	1.1
		National Grid SAP Adoption	1.0
		Organization Risk Management: The Widget Company	1.0
		Penetration Test	1.1
		Ransomware	1.1
		Secure LAN	1.1
	Law & Regulation	National Cybersecurity Governance	1.0
		Ransomware	1.1
	Human Factors	ACME Water	1.0
		FAA ERAM Outage	1.0
	Privacy & Online Rights	ACME Water	1.0
		Driver Assistance System Safety & Security	1.0
		Penetration Test	1.1
		Role Based Access Control	1.1

Attacks and Defences	Malware & Attack Technologies	Deciphering	1.0
		Mt. Gox Bitcoin Theft	1.0
		Penetration Test	1.1
		Ransomware	1.1
		Using Malware Analysis to Improve Security Reqs	1.1
		Wireshark	1.1
	Adversarial Behaviours	Heartland Payment System Breach	1.0
		Mt. Gox Bitcoin Theft	1.0
		Penetration Test	1.1
		Ransomware	1.1
	Security Operations & Incident Management	Heartland Payment System Breach	1.0
		Mt. Gox Bitcoin Theft	1.0
		National Cybersecurity Governance	1.1
		Penetration Test	1.1
		Ransomware	1.1
	Forensics	Mt. Gox Bitcoin Theft	1.0
		Wireshark	1.1
Systems Security	Cryptography	Deciphering	1.1
		Mt. Gox Bitcoin Theft	1.0
		Penetration Test	1.1
	Operating Systems & Virtualisation Security	Deciphering	1.0
		Heartland Payment System Breach	1.0
		Penetration Test	1.1
		Secure LAN	1.1
	Distributed System Security	Driver Assistance System Safety & Security	1.0
		Secure LAN	1.1
		Wireshark	1.1
	Formal Methods for Security	Deciphering	1.1
		Tokeneer ID Station Project	1.0
	Authentication, Authorisation & Accountability	ACME Water	1.0
		Heartland Payment System Breach	1.0
		Mt. Gox Bitcoin Theft	1.0
		Penetration Test	1.1
		Role Based Access Control	1.1
		Secure LAN	1.1
Software Platform Security	Software Security	Driver Assistance System Safety & Security	1.0
		FAA ERAM Outage	1.0
		Penetration Test	1.1
	Web & Mobile Security	Driver Assistance System Safety & Security	1.0
		Role Based Access Control	1.1
		Secure LAN	1.1
	Secure Software Lifecycle	ACME Water	1.0
		Aircraft Service Application	1.0
		Drone Swarm	1.0
		National Grid SAP Adoption	1.0
		Secure Acquisition	1.0
		SQUARE	1.0
		Tokeneer ID Station Project	1.0

		Using Malware Analysis to Improve Security Reqs	1.1
Infrastructure Security	Applied Cryptography	Deciphering	1.1
		Penetration Test	1.1
	Network Security	Role Based Access Control	1.1
		Secure LAN	1.1
		Wireshark	1.1
	Hardware Security	Driver Assistance System Safety & Security	1.0
	Cyber-Physical Sys Security	Driver Assistance System Safety & Security	1.0
	Physical Layer & Telecommunications	Penetration Test	1.1
		Secure LAN	1.1
		Wireshark	1.1

The collection of case studies provides a robust degree of coverage of the CyBOK knowledge areas and learning outcomes, especially in the fundamental topics of "risk management & governance" and "secure lifecycle management". We welcome and invite readers to contribute their case studies to provide a well-rounded library to help other software engineering instructors with teaching the CyBOK in their curriculum.

In addition to the library of case studies presented in this report, there are other collections of CyBOK-mapped teaching resources for educators and trainers available on the CyBOK website[1]. For example, Schreuders[2] presents a wide range of lab exercises related to the Network Security, Security Operations & Incident Management, Malware & Attack Technology, Adversarial Behaviour, Software Security, Authentication, Authorisation & Accountability, Operating Systems & Virtualisation, Forensics, Cyber-Physical Systems Security (CPS), Web & Mobile Security, and Cryptography.

4 Preliminary Objective Results from Applying the Case Studies

The complexities of software engineering and the competencies expected of software application developers are continually increasing. Central to building competencies is knowledge that must be organized, systematically communicated, and applied to real-world situations. Learning the requisite knowledge is critical for the security of an organization, however, educators have often struggled in understanding how learning occurs. To aid in this understanding, a variety of learning models have been developed along with measuring specific outcomes, setting threshold standards, and the development of learning frameworks [36]. The many learning models that have been developed provide the basis to help understand learning behaviors and ultimately to inform the design of instruction in the classroom. Real-world case studies have been instrumental and are often utilized to assist software engineers in obtaining requisite knowledge as well as to develop problem solving skills for projects they will encounter after graduation.

Since 1984, the Software Engineering Institute (SEI) has been committed to improving the practice of software engineering [30]. In an early effort to influence software engineering curriculum development throughout the education community, the SEI recruited a software educator to lead the effort. Workshops were conducted that included leading software engineering educators and practicing engineers to develop 'curriculum modules', which led to the curriculum guidelines that became the model curriculum at many universities [30]. An outgrowth of the curriculum project was the development of freely available

[1] https://www.cybok.org/resources_developed_through_funded_projects
[2] https://github.com/cliffe/SecGen/blob/master/README-CyBOK-Scenarios-Indexed.md

educational materials, which included case studies, to support faculty members teaching software engineering courses, presented in various workshops.
In addition to these workshops, curricula and educational materials were presented and discussed at the Conference on Software Engineering Education (CSEE). Feedback from faculty at the CSEE conference indicated that the case studies and examples were among the most useful materials, along with curriculum modules exploring a single software engineering topic, as they could be used directly in courses developed by faculty at their own universities. The full courses were seldom used directly by faculty as they naturally preferred developing their own courses once they became familiar with the material. Additionally, faculty members and industry trainers like the fact that the educational materials are structured so they could easily be tailored for international variations and incorporated into courses developed by faculty world-wide [30].

It is in this tradition that we developed the case studies presented in this library, specifically for the CyBOK curriculum. Many of the case studies presented herein have been applied for years in many SEI courses (e.g., SQUARE and Software Acquisition). Others have been designed specifically for this CyBOK library (e.g., Mt. Gox and Heartland Breach). Again, others have been derived from previous experience, such as the Driver Assistance System case study, which we describe exemplarily in the following.

4.1 Driver Assistance System Case Study

The Driver Assistance System case study is based on an industry-realistic case example provided by our industry collaborators during a publicly funded research project. The case study follows the approach in [19], [34] regarding its application and achievements. Results and experiences pertaining to student motivation and retention in a safety requirements engineering (RE) course are described in [37]. Yet, to meet ABET accreditation requirements, the RE course presented in [37] needed to be adapted after Spring 2019 to provide additional cybersecurity learning outcomes. This was done based on CyBOK, for which the Driver Assistance System Case Study was created. While the case study is freely available in the link provided in Section 3, we give a very brief overview here to frame the results from application, presented in Section 4.2.

Background. The case study describes the purpose of modern driver assistance systems (e.g., adaptive cruise controls or lane keeping support) and alleges that a nefarious hacker may be able to gain access to the car's safety-critical features through OEM-specific cloud-based connectivity systems (e.g., OnStar, BlueLink, meConnect, ConnectedCar, etc.).

Case Study Overview. The purpose of the case study is to familiarize students with the similarities and differences of safety and security requirements while building a complete, consistent, safe, and secure requirements specification consisting of natural language and model-based requirements as well as a safety argument and security assessment report.

Student Instructions. Students are asked to build the requirements specification based on the feature descriptions of car systems found in a real-world glovebox manual for a modern car. In three milestones (which in turn are subdivided into tasks), students will:

1. "Reverse engineer" the user requirements for one of the car's driver assistance systems and document goals and scenarios as well as natural language requirements in an IEEE 830-compliant requirements specification.
2. Conduct a Safety Hazard Analysis and Security Risk Assessment using a provided template and tutorial slides, derive safety mitigations as well as countermeasures, and document them as requirements in the specification from milestone 1.
3. Develop UML class, activity, and state machine diagrams to refine the requirements from milestones 1 and 2 to "a degree that would enable implementation" and develop a safety argument.

Instructor Notes. Instructors are advised that the case study at hand is intended as a semester-long team project for 2-5 students and recommends frequent team presentations of partial and preliminary solutions so other teams in the course can get exposed to solution alternatives.

Example Solution. There is no example solution for this case study, however, notes are presented that frame the degree to which aspects such as requirements completeness would allow for, e.g., hypothetical implementation.

References. This section contains references to the glovebox manual in question, hazard analysis template and tutorial, as well as some further reading.

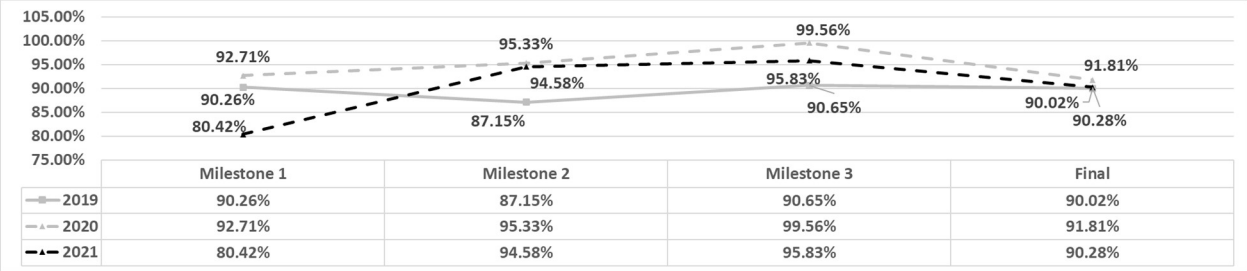
4.2 Results and Experiences from Applying the Driver Assistance System Case Study

As mentioned in Section 4.1, the RE course has employed industry-realistic case studies before [37] and was modified after 2019 with cybersecurity learning outcomes. Specifically, from Spring 2019 to Spring 2020, lecture units were added to convey risk and vulnerability analysis and modeling, and an exam question was added for formative assessment. However, in Spring 2020, the case study was not yet modified to foster CyBOK learning outcomes. Specifically, the case examples used in that semester largely mimicked the case study outlined in Section 4.1 yet lacked cybersecurity-related milestones and task.

In Spring 2021, the Driver Assistance Case Study described in Section 4.1 was used in addition to the lecture and exam assessment on cybersecurity that were added in Spring 2020. Assessment in the case study took the form of grading student solutions out of 15 points based on criteria such as correctness of used notations, consistency of information throughout the specification document, specificity (i.e., lack of vagueness and ambiguity), and completeness. Specificity and completeness in this sense mean that requirements should be specific and complete enough to allow hypothetical implementation, or alternatively highlight missing information to be determined in future hypothetical work (i.e., during system architecture design by another team, which was beyond the scope of the RE course).

Figure 1 shows the relative performance in all three case study milestones as well as the cumulative final grade for the 2021, 2020, and 2019 semesters, respectively. Note that to increase legibility, the vertical axis is scaled to the interval [0. 75..1.05]. As can be seen, project grades remained comparably high across all semesters and all milestones. Since project performance using case studies is typically at a very high level (cf. [34]), this seems to indicate that the specific instructions and structure of the Driver Assistance System Case Study led to a comparable performance. It is notable, however, that milestone 1 in 2021 was roughly ten percentage points below the 2019 reference semester and 12 percentage points below the previous year. We attribute this to students struggling with reading, comprehending, and deriving requirements from the car's real-world glovebox manual, instead of inventing requirements by themselves as in previous years.

Fig. 1 Project Performance in all three Case Study Milestones and Final Project Grades across Semesters



	Milestone 1	Milestone 2	Milestone 3	Final
2019	90.26%	87.15%	90.65%	90.02%
2020	92.71%	95.33%	99.56%	91.81%
2021	80.42%	94.58%	95.83%	90.28%

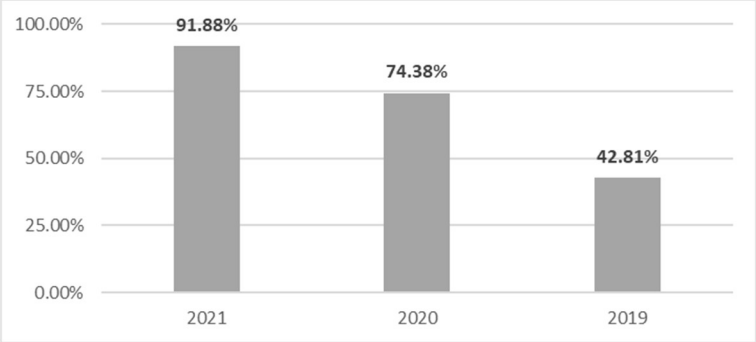
Nevertheless, we conclude from this that the Driver Assistance System Case Study was able to repeat the success from previous years' application of industry-realistic case studies in the safety requirements engineering course in question.

To assess whether this case study fostered CyBOK learning outcomes, we compare student performance in an exam question. As outlined above, in partial response to ABET accreditation requirements, lecture material as well as one exam question were added to the course for this purpose. The lecture material consisted of one intensive lecture on threats, attack vectors, and risk assessment which also contrasts safety engineering and security engineering principles with one another. Furthermore, security-related lecture material was interspersed with already-existing lectures, e.g., misuse case modeling as part of scenario-based RE and threat modeling during safety argument construction. The exam question required students to synthesize knowledge on safety analysis and cybersecurity risk assessment by presenting them with a Functional Safety Hazard Analysis template (which they were already familiar with from working on the case study) and evaluating what needed to be changed to accommodate concepts such as "threat", "risk", "vulnerability", and "countermeasure."

Figure 2 shows the difference in the cybersecurity assessment score in the final exam (i.e., the average score achieved by all students in the cybersecurity-related question on the exam) across all three semesters. Please note that while in 2019, only minimal security-related instruction took place in the course, a similar question requiring concept synthesis existed in the final exam. Albeit this is not comparable to the assessment in 2020 or 2021, we present the score in Fig. 2 for reference. In 2020 and 2021, both courses were identical, except that in 2021 the CyBOK case study was applied.

Unsurprisingly, students performed much better in security-related assessment in 2020 onward. Since the CyBOK-related curriculum between 2020 and 2021 differed only in application of the Driver Assistance System Case Study, the 17.5 percentage point increase from the 2020 and 2021 semester must be attributed to it. To test whether this increase is significant, we conducted a T-Test to verify differences in means (after rejecting the assumption of variance equality by means of an F-Test). Results are shown in Table 2.

Fig. 2 Cybersecurity Assessment Score in Final Exam



T-Test results reveal that the difference between the 2021 and 2020 semester is significant ($p < 0.05$). Since the mean for the 2021 offering is higher (91.88% vs. 74.38%), we reject the null hypothesis and accept these results as evidence that the Driver Assistance System Case Study significantly increased students' CyBOK learning outcome (as it pertained to the case study). A post-hoc power analysis using Cohen's d [38] revealed a medium-large effect size ($d = 0.706$), indicating a low likelihood of statistical error to cause significance (despite the low sample size, $\alpha = 0.037\%$).

Table 2 Analysis of Cybersecurity Assessment Score between 2020 and 2021 (2019 shown for Reference)

	2021	2020	2019
Mean	91.88%	74.38%	42.15%
Variance	20.38%	28.50%	20.86%
Sample Size	16	16	31
dF	27		
F	0.5113 (unequal variances)		
Student's T	0.0318		
Cohen's d	0.706 (medium-large effect size)		

Finally, we would like to share some qualitative experience regarding the Driver Assistance System Case Study. Our experience throughout the semester mimicked experiences reported in related work (see, e.g., [19], [34], [37]). In particular, we noticed a steep learning curve regarding safety-related concepts. However, even though the core concepts of safety and cybersecurity are comparatively relatable, students seemingly struggled less in finding, e.g., threats as opposed to hazards. Cybersecurity concepts seemed to be almost intuitively understandable, while safety concepts were not.

One of the motivating factors of assigning the Driver Assistance System Case Study as a project was that students would pick different vehicle systems and, during the frequent in-class presentations, would actively exchange ideas and recognize logical interfaces between driver assistance systems (for example, the team working on the Lane Keeping Assist system would realize that the forward-facing camera can also be used in the adaptive cruise control system, hence leading to collaboration between the respective teams during safety analysis and threat modeling). Despite the instructor's best efforts to point out similarities across projects and encouraging cooperation beyond class discussions, student reactions rarely exceeded acknowledgement that certain interfaces are similar ("Oh, yeah, we have a camera, too"). Instead, students focused on producing their own isolated solution. A confounding factor may have been the mode of interaction, as due to the COVID-19 pandemic, the RE course had to resort to synchronous online instruction using video conferencing throughout 2021 and partially in 2020.

5 Subjective Case Study Classroom Experiences

In addition to the data collected in section 4, a number of case study authors provided subjective feedback on the results of case study usage in the classroom, in both academic and industry/government organizational settings. While subjective feedback is not evidence, it adds to the body of knowledge on classroom use of the case studies, it was generously provided by the case study authors and is included here.

5.1 Acme Water (by Shamal Faily)

5.1.1 About the study

This case was developed around a specification exemplar [40] of a UK Water Company. The exemplar is based on anonymised data from an actual water company that was the subject of past studies, e.g. [41], [42], [43]. The exemplar has features that make it useful for teaching security. First, the exemplar challenges student preconceptions. Operational technology may look like at ICT they are familiar with, but the security and privacy threats and vulnerabilities associated with such technology is different. Second, because the exemplar is based on a real organisation, instructors can use their own experience of the case to provide additional context when reviewing student or worked solutions to exercises.

5.1.2 Module and Student background

The case was designed to support a module on *Security by Design*. The accompanying text for this unit [44] also introduces and applies the ACME Water exemplar.

The module is taken by final-year BSc (hons) Forensic Computing & Security students, and MSc Cyber Security & Human Factors students. The combined cohort is typically between 80-100 students. The exercises are carried out in classes of 15-20 within small groups of 4-5 students.

The exercises were designed for on-campus delivery. However, with only minor modifications to the requirements, were also delivered remotely via Zoom during the COVID-19 pandemic.

The experience and confidence level of both cohorts in underpinning content in software engineering, HCI, and cybersecurity varies. The undergraduates are typically comfortable with cybersecurity concepts, but unfamiliar with software design and HCI concepts exposed to them in previous years of their course. The backgrounds of the postgraduates are more variable; they are usually less confident in

Cybersecurity and Software Engineering but motivated to learn. Experience and motivation in HCI are even more variable for the postgraduates; they range from limited for computing and engineering graduates to strong for social science and humanities graduates.

5.1.3 Experiences

Drawing on personal and anecdotal experience using the case for four years, three themes summarise the challenges faced by students when working with the case.

System modelling. Students have difficulty modelling systems. For example, the Introduction exercise was designed to re-acquaint and, in some case, acquaint students with two fundamental software design models: UML class diagrams and use cases. It was designed to demonstrate how lightweight but precise models can flesh out ambiguity and identify potential sources of vulnerability and human error. At the outset of the exercise, many students struggle with simultaneously using the modelling techniques and comprehending the problem description. The experiences with the Risk & Trust and Privacy exercise, which entail modelling DFDs, have been similar. However, these struggles subside as students gain confidence and begin to identify system vulnerabilities.

Critical reflection. Several of the exercises require students to apply the content related to the exercise to find problems with a proposed design. Students sometimes find doing this a challenge. For example, in the Risk & Trust exercise, students would often 'low hanging fruit' such as the use of HTTP (rather than HTTPS) but would need prompting to spot ambiguity in the interface design, or the unwarranted use of trust symbols. Similarly, in the Security Economics & Entrepreneurship exercise, students occasionally require guidance on the different individual costs, and need to be challenged on the idea that policy proposed is not intrinsically good or bad, but needs adapting based on other controls.

Working with security complexity. Students also faced difficulties comprehending the complexity resulting from security because of implicit and unwarranted assumptions. This is best illustrated by the Authorisation exercise, which draws out the challenges of designing security for infrastructure that can only be indirectly perceived. Students often start the exercise with pre-conceived ideas that make it hard to appreciate who the different stakeholders are. The exercise becomes more challenging when they are asked to adapt a user interface for a proposed role-based access control solution but fail to recognize the user interface design is based on a discretionary access control model. In the Architecture exercise, students similarly struggle to comprehend the security implications of the architectural model, but -- once the attack surface metrics were applied - - the architectural risks became apparent.

5.1.4 Benefits to students

The case was successful in reinforcing the merits of software design and engineering when addressing Cyber Security problems. The case was also successful in showing how user research and techniques from HCI could draw out security problems, and how students' own unwarranted assumptions could contribute to these problems. Such lessons were not always welcomed by students. A small number of students over the years have, when completing mid-unit student feedback, claimed it would be unlikely such knowledge would be useful to them for their future careers. Although we do not track the careers of all those who used this case, we have anecdotal feedback from some students who found themselves using the course material related to this case a few years after graduating.

5.2 GPS Spoofing of UAV (by Carol Woody)

This case was developed using publicly available information about a specific incident reported by news media. Security experts surmised the details of how such a result of spoofing could have occurred. The case was used in several technical discussion session with program managers, designers, and engineers, who were not trained in cybersecurity, to help them understand how choices made in design could contribute to a cyber incident. There were 10-15 in each session.

The initial reaction of the participating designers and engineers in the various sessions was anger that the incident was being discussed since they indicated many of the choices are made as part of a conceptual design over which they had limited control. They felt threatened with blame for something they insisted was out of their control. As we talked further, they began to understand that, at a minimum, they were responsible for notifying those making these decisions of the level of risk they were taking.

The program managers were asking how they could better identify these issues during design and how they could determine design flaws in an acquisition. For those we discussed the criticality of off-nominal testing to ensure a range of possible behaviors were considered

5.3 Organization Risk Management: The Widget Company Case Study (by Carol Woody)

This case was developed to support the delivery of public training classes for the **Operationally Critical Threat, Asset, and Vulnerability Evaluation℠** (OCTAVE®). Students came into the course with a wide variety of knowledge levels and exposure to operational risk identification and management. Course offerings were capped at 25 participants and required 2.5 days to complete. The widget company provided a neutral, target rich environment for student use. There are eight sections to OCTAVE and students were typically divided into groups of 5-7 to work together in each of the steps and present their results to the group. For those with little background with security risk, this material was "eye opening". Many were able to identify existing practices in their own organization that represented unacceptable risk. Few were able to make the changes needed to mitigate the risks which allowed us to discuss how such issues should be escalated and whether their organization was prepared to consider them appropriately.

The materials were also used in workshops delivered onsite for specific organizations. Participants came from technology using segments of the organization to discuss cyber risk challenges. This was a one-day session. The example solutions were displayed and discussed with the group to compare their current practices with the issues identified at Widget Company. In one instance a marketing manager for a manufacturing company left the room quickly to call one of his negotiating teams to make sure they included consideration of security risks in a contract they were negotiating with a foreign subcontractor.

5.4 Penetration Test Case Study (by Bastian Tenbergen)

This case study has been developed as part of a semester-long final project in a cybersecurity course at SUNY Oswego. At the time of writing this report, the case study has been successfully applied in five sections over three years, following the grading scheme suggested in the instructor notes.

Experience shows that roughly 1/3 of learners will successfully penetrate at least one target system and finish early. These students were encouraged to attempt to penetrate another system of their choice, but only a minority followed suit. Another 1/3 of students typically finish by the end of the project, mostly in the last week. The final 1/3 of students will either give up (approx. 10% of all students) or be unsuccessful. Approximately 15% of all students may find and refer to a walkthrough published online, which they found using clues about the vulnerable systems (e.g., the search string "penetration test vegeta" leads to the walkthrough of one of the virtual machines on YouTube, which students were allowed, even encouraged to use). Even students who did not successfully penetrate a system typically find the experience to try very valuable, as evidenced by the journals they are compelled to keep and the penetration test report they are required to submit. Especially the reports are an invaluable tool to allow students to meet learning outcomes regarding security management and incident reporting.

Students come with a variety of backgrounds and about 1/3 to half of the students have limited to no command line experience in Linux operating systems. Penetration testing should therefore include a mini-lecture or selected resources on using the command line. About ¼ of students typically gets stuck at the very beginning, i.e., right after the "getting started" instructions end. This is mostly due to a lack of imagination when it comes to assessing threat vectors and trying different modes of access. In particular,

even when students are repeatedly encouraged to scan for ports and googling results, a sizable number of students will not do this. For this reason, we have begun offering students a digital discussion forum alongside the project to help each other out and collect ideas from each other. This is fairly successful in most sections, allowing students with limited Linux background to continue the project.

A fairly common occurrence is that many students rely on other students' progress. This takes two forms: (1) using the tool "jacktheripper" to crack another student's user account on the same machine or (2) using the tool "wireshark" to read network traffic, hoping to intercept a password. This is insofar not ideal as both strategies are unlikely to be successful: (1) relies on other students storing information in their Kali account, which rarely happens. In addition, using the tool is possible, but not required, as all students have root access on Kali anyways and could therefore simply force themselves into other students' accounts. (2) entails observing a successful hack as it happens, however given the project length and aptitude of other students, this possibility is remote at best. Instead, these students also benefit from the discussion forum mentioned above and explanations of other students' approaches.

Instructors are strongly encouraged not to associate case study grade with penetration success. Instead, full scores should be awarded to students who demonstrate active engagement and an honest effort in applying penetration techniques, even if they are unable to successfully achieve root access on at least one target machine.

5.5 Secure Acquisition Case Studies (by Dan Shoemaker and Anne Kohnke)

The following describe four Lab Assignments that are part of a Secure Acquisition Course at University of Detroit Mercy. The text in this section is extracted from [10]:

Lab 1: Project Initiation – The student will define the potential system functions required for a given application (in a case study provided), as well as who will supply them. The following steps are required:

1. Identify a business process in the case that will be supported by a new system.
2. Define top-level system functionality required to perform that process (these must be coherent, e.g., logically related and complete).
3. Decompose top-level system functionality into a second level of component functions,
4. Decompose the second level functions into a third level (e.g., formulate a component tree). Assign a supplier for each component at all tiers – these are the potential supplier organizations listed in the overall case assignment. They are assumed to be the potential subcontractor organizations.

Lab 2: Acquisition/SCRM Project Risk Analysis – The student will use the assessment tool provided in the case to evaluate the general risk status of the supplier community. Then mitigation strategies and timelines will be developed to address all potential risks. To do this, the student will utilize the forms taken from 12207/NISTIR 7622 (provided). The complete list of potential base practices is listed there. To facilitate this process, it is useful to create a generic checklist from the specifications of ISO 12207/NIST SP800-161 to identify the risks that might affect components, or participants. Using this list, identify and assess risks against the base practices for the potential suppliers in the case's supplier list in order to obtain an overall risk rating. The risk rating will be used to prioritize the risks.

Lab 3: Adequacy of Current Acquisition Practice – Using the checklist, created in Lab 2, evaluate the current capability maturity of your organization's acquisition process as stated in the case. The assessment will determine areas where proper acquisition risk management is being practiced, as well as the relative maturity of those practices. The goal is to generate a nominal ranking of capability maturity based on a common scale of process performance.

Lab 4: Supplier Capability Evaluation - Using the checklist, assess the competency of each individual contractor organization in the supply chain. You must provide a complete plan and timeline for doing this. For each practice please rate (1-3) its execution as:

1. Incomplete: The Incomplete level has no common features,

2. Performed: Base practices of the process are generally performed. The performance of these base practices is ad-hoc and is not rigorously planned or tracked,
3. Managed: The performance of the process is planned and tracked. Base practices are performed according to a well-defined process using approved methods.

Evaluation of the final project report is based on a rating of (1-3):

1. Conformity with the principles of secure acquisition management best practice (presented in lecture),
2. Complete and correct tailoring of the requisite activities of the ISO 12207 and NIST SP 800-161 principles to the particular case, and
3. A complete, correct, and unambiguous presentation of findings. Projects are assessed based on the following criteria (relative weights in parentheses).

All work products are rated based on their demonstrated level of the following elements and given a point value. The maximum number of points for each element are as follows: Correctness (5), Completeness (5), Unambiguousness (5), Understandability (6), Modifiability (3), Traceability (3), and Annotation (3).

The rest of the grade involves a final cumulative examination for the course.

Seven Years of Experience. Since a formal body of content and a practical educational approach did not exist in 2012, the Department of Defense (DoD-CIO) contracted with the Institute for Defense Analysis (IDA) to create a common course package for the teaching of methods for secure acquisition in general higher education. UDM faculty did some of the initial work and the Secure Acquisition course presented here was developed directly from that research.

Since 2013, the Secure Acquisition course has been offered at the University of Detroit Mercy once per academic year. The first three offerings were conducted in a traditional classroom setting. The last four have been delivered in an all-online format. The delivery and project work of the students who took the class online, was consciously structured to mirror the work that was delivered in the on-ground classroom format. It is a required course in the Master of Science in Information Assurance/Cybersecurity degree program (MS-IA), which has been a continuously designated NSA CAE/IAE (now CAE/CD) for the past fifteen years.

The MS-IA student body is highly diverse, with a large percentage of women and veterans. The average student is in their early thirties. It is primarily comprised of working professionals from as far away as Germany. However, ninety six percent of the students currently reside in Michigan and neighboring states.

The field of cybersecurity is still too new to have reliable standardized tests. So, the students are individually admitted to the program based on GPA as well as intangibles such as military service. There have been occasional students with bachelor's degrees in cybersecurity. However, the assumption has been that entering students are only proficient in one aspect of the field, such as software engineering. So, the approach has been to treat the topic of secure acquisition as if it was a novel concept.

At present, the secure acquisition course is one of the hottest tickets in the entire program. The student body of the MS-IA is intentionally kept small and elite. Still, for the past two years, the course has been at its maximum capacity, which is twenty-five students. It had a waiting list last year. Previously, new student enrollments numbered: twelve (2013), eleven (2014), sixteen, (2015) and eighteen (2016). The steady increase in enrollment is partly due to the transition to the all online format. However, it also reflects the growing awareness of the need for secure ICT acquisition in the automotive supply chain. This has been a stated reason why one of the Big Three and a local energy company underwrite enrollment of their employees in the MS-IA program. Since we presume there is no original baseline for comparison, there has been no pre-post testing of performance. However, we do use mean standard deviation as the basis for grading. In essence, the students are given numeric grades and their outcomes fall along a statistical distribution. We give five points for the individual lab assignments, thirty points for the final project and examination; for a total of eighty points. Table 3 below displays the final class average scores. The following should be noted in reading this. Since these are individually selected candidates, the student

body is highly competitive and very homogenous. Many of the students are currently employed in project manager positions. Thus, the fact that the curve would be skewed as much as it is to the higher end of the scale, could be expected. The course packages process considerations into a commonsense approach.

Table 3. Final Class Averages for CYBE 5470 Secure Acquisition

Final Class Averages for CYBE 5470 Secure Acquisition		
Year	Mean	Standard Deviation
2013	72	4.00
2014	74	3.00
2015	71	4.50
2016	71	4.50
2017	75	2.50
2018	74	3.00
2019	76	2.00

The jump in performance between the 2016 and 2017 academic year can be attributed to the adoption of the all online delivery format, which seems to resonate better with older working students. Almost all courses in the program have experienced the same increase in student performance, which has caused us to reevaluate our overall grading process. The other reason for the increase can be attributed to the immediate, hands-on nature of the lab presentation. Prior to 2017, all of the lab work was done asynchronously. The synchronous "walking through" of the lab assignments, as part of the weekly recitation lectures, has been effective in helping students understand good acquisition security practices. For instance, a requisite specification of requirements, which is a common artifact in the development process, is also a critical element in contract formulation, since it dictates how acceptance will be evaluated. Most of the students understand how specification works. They also understand contracts. However, the critical link between the functions required and the need to fully capture them in contractual terms is not always evident. It is the ability to explain the relationship at the point where the question arises that makes the teaching model so effective.

5.6 SQUARE Case Study (by Nancy Mead)

The SQUARE Case Study was used in a wide variety of continuing education training courses and workshops, in academic Master's level courses, and in faculty train-the-trainer courses. This occurred over a period of more than 10 years. In some cases, students were allowed to apply the SQUARE method to a case study of their choice, rather than using the version of the case study provided on the CyBOK website. Objective data was not collected on the influence of the case study on student learning. Students who did well on the case study, generally did well in courses where SQUARE was taught, but they may have been better students to begin with.

Feedback on the case study part of the courses was generally favorable. Many students indicated that using the case study to practice the SQUARE method gave them a much better understanding of the method itself than they would have had otherwise. Students who applied SQUARE to a case study of their choice, usually an assignment from another course, indicated that it helped them in the other course. However, some students indicated that when they proposed adding security requirements to the case studies in their other courses, they often got nowhere because of development cost and schedule issues. Since students often work on the development of a project, and not on its maintenance during operational usage, they don't see the cost benefits over the full product lifecycle. This is reflective of what we see in practitioner organizations, past and present.

Government attendees who were able to specify security requirements, appreciated the benefit that they got from the case study. In some cases, they focused on security requirements for acquisition rather than development, which of course is equally as important.

One difficulty that was encountered early on, was that students and faculty who were not native English speakers needed more time than was allocated in order to read and understand the case study materials, which were all in English. Once this issue was recognized, the time allocated for each of the SQUARE steps in the case study was increased. We also allowed the students to execute the steps using their native language and translate the final results into English. Additionally, some of the SQUARE papers, reports, and course materials were translated into other languages by faculty who intended to teach the method.

At one point, the SQUARE methodology lectures were captured on video for an SEI certificate program in Cybersecurity Engineering. The SEI instructional designers supporting the work did an outstanding job of making the case study more accessible for distance learning students.

6 Conclusion and Future Work

In this report, we presented a library of 25 case studies for the Cybersecurity Body of Knowledge (CyBOK), version 1.1 (2021). The work was supported by The UK National Cyber Security Centre [1], [17]. Case studies were developed by a team of subject-area expert authors. We presented initial favorable objective results from the application of one case study in a Safety Requirements Engineering course that heavily emphasizes cybersecurity. These results show that the use of the Driver Assistance Case Study had a significantly positive impact on learning outcomes as assessed by a final exam. We also presented a substantial number of subjective results from use of other CyBOK case studies in the classroom. These results reinforce the evidence of case study benefit in formal course work.

The purpose of the creation of the case study library was to provide sample educational materials for instructors to educate the next generation of cybersecurity software engineering professionals in CyBOK's topic categories and knowledge areas. The library consists of 25 case studies, which share a common structure for expedient and easy adoption, including context, student instructions, instructor notes, and sample solutions. All CyBOK knowledge areas are covered, in many cases with multiple case studies, allowing for variety in instruction such as application as group projects or as exam questions.

In the future we plan to continue to collect quantitative data to support improved learning outcomes and we encourage others to join us and contribute to this important data collection effort.

Acknowledgements

The authors appreciate the kind funding from The UK National Cyber Security Centre to facilitate development of the case studies presented herein. Much of the general content in the Abstract and Sections 1 through 4 in this report is extracted from [39]. We thank Shamal Faily, Anne Kohnke, Dan Shoemaker, and Carol Woody for their additional contributions on classroom usage in this report.

References

[1] The National Cyber Security Centre, The Cyber Security Body of Knowledge (CyBOK), Version 1.1. © Crown Copyright, 2021, UK Open Government License. Accessed 12/31/2021, available at: https://www.cybok.org/

[2] Chowdhury, N., Adam, M., Skinner, G., The Impact of Time and Pressure on Cybersecurity Behavior: A Systematic Literature Review. Behavior & Information Technology 38(12), 2019, pp. 1290-1308.

[3] Krasner, H. The Cost of Poor Software Quality in the US: A 2020 Report. Consortium for Information & Software Quality. 2021. https://www.it-cisq.org/pdf/CPSQ-2020-report.pdf

[4] O'Driscoll, A. 25+ cyber security vulnerabilities statistics and facts of 2021. Comparitech, 2021, https://www.comparitech.com/blog/information-security/cybersecurity-vulnerability-statistics/

[5] Shaw, M., Software Engineering Education: A Roadmap. In Proc. Future of Software Engineering, 2000, pp. 371-380.

[6] Sedelmaier, Y., Landes, D., Systematic evolution of a learning setting for requirements engineering education based on competence-oriented didactics. In Proceedings of the IEEE Global Engineering Education Conference, 2018, pp. 1062–1070.
[7] Gabrysiak, G., M. Guentert, R. Hebig, and H. Giese, Teaching requirements engineering with authentic stakeholders: Towards a scalable course setting. In Proceedings of the First International Workshop on Software Engineering Education Based on Real-World Experiences 2012, pp. 1–4.
[8] Mishra, D., Ostrovska, S., Tuna, H., Exploring and expanding students' success in software testing, Information Technology and People 30(4), pp. 927-945, 2017. DOI:10.1108/ITP-06-2016-0129.
[9] Sonatype, 2020 State of the Software Supply Chain: The 6th annual report on global open-source software development, Fulton, MD. https://www.sonatype.com/hubfs/Corporate/Software%20Supply%20Chain/2020/SON_SSSC-Report-2020_final_aug11.pdf
[10] Shoemaker, D., Mead, N., Kohnke, A., Teaching Secure Acquisition in Higher Education. IEEE Security & Privacy 18(4), pp. 60-66, 2020.
[11] Bosch, J., Speed, Data, and Ecosystems: The Future of Software Engineering. IEEE Software 33(1), 2015, pp. 82-88.
[12] Harlen, W., James, M., Assessment and Learning: Differences and Relationships between Formative and Summative Assessment. Assessment in Education: Principles, Policy & Practice 4(3), 1997, pp. 356-379.
[13] Ramirez, R., Choucri, N., Improving Interdisciplinary Communication with Standardized Cyber Security Terminology: A Literature Review. IEEE Access 4, 2016, pp. 2216-2243.
[14] Bourque, P., Fairley, R., Guide to the Software Engineering Body of Knowledge (SWEBOK ®), Version 3.0. IEEE Computer Society Press, 2014.
[15] Švábenský, V., Vykopal, J., Čeleda, P., What are Cybersecurity Education Papers About? A Systematic Literature Review of SIGSCE and ITiCSE Conferences. In Proceedings of the 51st ACM Technical Symposium on Computer Science Education, pp. 2-8, 2020.
[16] Joint Task Force on Cybersecurity Education: Cybersecurity Curricula 2017: Curriculum Guidelines for Post-Secondary Degree Programs in Cybersecurity Education. accessed 5/27/21, available at: https://www.acm.org/binaries/content/assets/education/curricula-recommendations/csec2017.pdf
[17] Rashid, A., Chivers, H., Danezis, G., Lupu, E., Martin, A. (Eds.), The Cyber Security Body of Knowledge. © Crown Copyright, The National Cyber Security Centre, 2021. Accessed 12/31/21, available at: https://www.cybok.org/media/downloads/CyBOK_v1.1.0.pdf
[18] Man Sze Lau, A., Formative good, summative bad? – A Review of the Dichotomy in Assessment Literature. Journal of Further and higher Education 40(4), 2016, pp. 509-525.
[19] Daun, M, Salmon, A., Tenbergen, B., Weyer, T., Pohl, K., Industrial case studies in graduate requirements engineering courses: The impact on student motivation. In Proceedings of the IEEE 27th Conference on Software Engineering Education and Training (CSEE&T), 2014, pp. 3–12.
[20] Sivan, A., Wong Leung, R., Gow, L., Kember, D., Towards more active learning in hospitality studies, Intl. Journal of Hospitality Management 10(4), 1991.
[21] Bonwell, C., Eison, J., Active Learning: Creating Excitement in the Classroom, ASHE-ERIC Higher Education Report No. 1, The George Washington University, 1991.
[22] Richardson, I., Delaney, Y., Problem Based Learning in the Software Engineering Classroom, In Proceedings of the 22nd IEEE Conference on Software Engineering Education and Training (CSEE&T), 2009, pp. 174-181.
[23] Berry, D., Kaplan, C., Planned programming problem gotchas as lessons in requirements engineering. In Proceedings of 5th International Workshop on Requirements Engineering Education and Training, pp. 20–25, 2010.

[24] Rusu, A., Russell, R., Cocco, R., Simulating the software engineering interview process using a decision-based serious computer game. In Proceedings of the 16th International Conference on Computer Games (CGAMES), pp. 235–239, 2011.
[25] Marutschke, D. M., Kryssanov, V., Brockmann, P., Teaching distributed requirements engineering: Simulation of an offshoring project with geographically separated teams. In Proceedings of the IEEE 32nd Conference on Software Engineering Education and Training (CSEE&T), pp. 1–5, 2020.
[26] Grubb, A., Four Opportunities for SE Ethics Education. In Proceedings of the IEEE/ACM 2nd International Workshop on Ethics in Software Engineering Research and Practice (SEthics), 2021.
[27] Garg, K., and Varma, V., "A Study of the Effectiveness of Case Study Approach in Software Engineering Education", 20th Conference on Software Engineering Education Training (CSEET'07), (2007), 309–316.
[28] Kimball, B.A., The Emergence of Case Method Teaching, 1879s-1990s: Search for Legitimate Pedagogy, Bloomington, IN: Poynter Center.
[29] Runeson, P., Höst, M., Guidelines for Conducting and Reporting Case Study Research in Software Engineering. Empirical Software Engineering 14, 131, 2009. https://doi.org/10.1007/s10664-008-9102-8
[30] Druffel, L., A Technical History of the SEI. Special Report CMU/SEI-2016-SR-027, Software Engineering Institute, January 2017. Accessed 5/27/21, available at: https://resources.sei.cmu.edu/asset_files/SpecialReport/2017_003_001_485151.pdf
[31] Saurabh Tiwari. Impact of cbl on student's learning and performance: An experience report. In Proc. of 13th Innovations in Software Engineering Conf. (ISEC 2020), ISEC 2020, New York, NY, USA, 2020. Association for Computing Machinery.
[32] P. Manohar, S. Acharya, P.Y. Wu, A.A. Ansari, and W.W. Schilling, Jr. Case study based educational tools for teaching software V&V course at undergraduate level. In122nd ASEE Annual Conf. and Exposition: Making Value for Society, 2015.
[33] N.R. Mead and E.D. Hough. Security requirements engineering for software systems: Case studies in support of software engineering education. In Proc. of 19th Conf. on Software Engineering Education and Training, volume 2006, pages 149–156, 2006.
[34] M. Daun, A. Salmon, T. Weyer, K. Pohl, and B. Tenbergen. Project-based learning with examples from industry in university courses: An experience report from an undergraduate requirements engineering course. InProc. of IEEE 29th Int. Conf. on Software Engineering Education and Training (CSEE&T), pages 184–193, 2016.
[35] B. Penzenstadler, M. Mahaux, P. Heymans, "University meets industry: Calling in real stakeholders", Proc. 26th IEEE Conf. Soft. Eng. Education & Training, 2013, pp. 1-10.
[36] Voorhees, R.A., Competency Based Learning Models: A Necessary Future, New Directions for Institutional Research, 2001, No. 110, Summer, John Wiley & Sons, Inc.
[37] B. Tenbergen, M. Daun, "Industry projects in requirements engineering education: Application in a University Course in the US and Comparison with Germany," In Proceedings of the Hawai'i International Conference on System Sciences, 2019.
[38] J. Cohen, Statistical Power Analysis for the Behavioral Sciences (2nd ed.), Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers, 1988.
[39] A. Kohnke, B. Tenbergen, N.R. Mead, "Using Cybersecurity Body of Knowledge (CyBOK) Case Studies to Enhance Student Learning", In Proceedings of the Hawai'i International Conference on System Sciences, 2022.
[40] Feather, M. S., Fickas, S., Finkelstein, A., and van Lamsweerde, A. Requirements and specification exemplars. Automated Software Engineering 4, 4 (1997), 419–438.

[41] Atzeni, A., Cameroni, C., Faily, S., Lyle, J., and Fléchais I. Here's Johnny: A Methodology for Developing Attacker Personas. In Proceedings of the 6th International Conference on Availability, Reliability and Security (2011), pp. 722–727.
[42] Faily, S., and Fléchais, I. Barry is not the weakest link: Eliciting secure system requirements with personas. In Proceedings of the 24th BCS Interaction Specialist Group Conference (2010), BCS '10, British Computer Society, pp. 124–132.
[43] Faily, S., and Fléchais, I. User-centered information security policy development in a post-Stuxnet world. In Proceedings of the 6th International Conference on Availability, Reliability and Security (2011), pp. 716–721.
[44] Faily, S. Designing Usable and Secure Software with IRIS and CAIRIS, 1st ed. Springer, 2018.