LUT University
School of Engineering Science
Software Engineering
Master's Programme in Software Engineering and Digital Transformation

Lassi Riihelä

TEACHING INFORMATION SECURITY – A SYSTEMATIC MAPPING STUDY

Examiners : Professor Jari Porras
D.Sc. (Tech.) Janne Parkkila

Supervisors: Professor Jari Porras
D.Sc. (Tech.) Janne Parkkila

ABSTRACT

LUT University
School of Engineering Science
Software Engineering
Master's Programme in Software Engineering and Digital Transformation

Lassi Riihelä

Teaching information security – a systematic mapping study

Master's Thesis

64 pages, 5 figures, 8 tables, 1 appendix

Examiners: Professor Jari Porras
D.Sc. (Tech.) Janne Parkkila

Keywords: teaching information security, information security, systematic mapping study

In this research a systematic mapping study is presented that investigates how the teaching of information security has been studied by the academia, identifying recommendations and guidelines. Four categories were identified from the articles: laboratory environments, teaching specific topics, how to teach information security, and course descriptions and teaching experiences. According to research, Information security is recommended to be taught using virtualized laboratory environments, utilizing the offensive approach, and it should be spread across the computer science curriculum. A suggestion is presented on how to take the recommendations and guidelines into account at LUT University's computer science education.

TIIVISTELMÄ

LUT-Yliopisto
School of Engineering Science
Tietotekniikan koulutusohjelma
Master's Programme in Software Engineering and Digital Transformation

Lassi Riihelä

Tietoturvan opetus – systemaattinen kirjallisuuskatsaus

Diplomityö

2019

64 sivua, 5 kuvaa, 8 taulukkoa, 1 liite

Työn tarkastajat: Professori Jari Porras
TkT Janne Parkkila

Hakusanat: tietoturvan opetus, tietoturva, systemaattinen kirjallisuuskatsaus
Keywords: teaching information security, information security, systematic mapping study

Tässä tutkielmassa esitetään kirjallisuuskatsaus, jonka tarkoituksena on selvittää miten tietoturvan opetusta on tutkittu, tunnistaen suosituksia ja ohjenuoria. Tutkimuksessa havaittiin, että tietoturvan opetusta on tutkittu neljästä eri näkökulmasta: Laboratorioympäristöjen käyttö opetuksessa, spesifien aiheiden opetus, miten opettaa tietoturvaa, sekä kurssikuvaukset ja opetuskokemukset. Aineiston perusteella tietoturvaa suositellaan opetettavan käyttäen virtualisoituja laboratorioympäristöjä tehtävien suorittamiseen, sekä suunnitelemaan opetus niin, että siinä tulisi esiin hyökkääjän näkökulma ja hyökkääminen. Lisäksi tietoturvan tulisi olla läsnä koko tietotekniikan koulutusohjelman opetuksen. Tutkielmassa ehdotetaan miten suosituksia ja ohjenuoria voi hyödyntää Lappeenrannan teknillisen yliopiston tietotekniikan opetuksessa.

ACKNOWLEDGEMENTS

Firstly, I would like to thank Jari Porras for having an endless amount of patient with me and this thesis process. Secondly, I would also like to thank Uolevi Nikula for encouraging me to finish this thesis, and Janne Parkkila for all the help. Erno, thank you for your help during a moment of despair.

To my family, especially to my father and mother, thank you for all the support you have given me. You will probably see a lot more of me now. Katja, thank you for understanding and letting me work many evenings on this thesis while you've had to take care of Hippi and the house.

To all my friends, I wish to apologize the fact this process took so long. Thank you for letting me put my studies first. I promise, from now on I will have more spare time.

Special thanks to Joni Herttuainen for being the tin in my foil, for being the "yarrr" in yarrrgon. Thank you for showing me how it's done and being an awesome friend. It's time to drink all the booze, and hack all the things.

Lappeenranta, May 30th, 2019

Lassi Riihelä

TABLE OF CONTENTS

APPENDICES

LIST OF SYMBOLS AND ABBREVIATIONS

ACM	Association for Computing Machinery
API	Application Programming Interface
CTF	Capture The Flag
CWE	Common Weakness Enumeration
DOS	Denial of Service
IEEE	Institute of Electrical and Electronics Engineers
MOOC	Massive Online Open Course
OWASP	The Open Web Application Security Project
RCE	Remote Code Execution

1 INTRODUCTION

Software systems are all around us. We interact with software daily in our lives, we browse the web, pay our bills in online banks, stream videos, and navigate to places using our mobile phones. It even helps us run parts of our critical infrastructure. However software is not without it's problems, the current state of the security of our software systems can be questioned. Data breaches are in the news non-stop, a hacker has managed to hack into a company's systems and steal sensitive data. New vulnerabilities are constantly found from old and new software systems, and those vulnerabilities can be used to attack the systems and steal sensitive data. WannaCry ransomware is one example of a cyber attack that utilized a vulnerability in a software system [1].

Vulnerability is a flaw in a software system, a flaw that can be exploited to take control of the attacked system. These flaws can exists in many places of the software systems, in the configuration, in the source code, or in both. Software systems are designed and developed by software engineers. It's the engineers who mistakenly introduce these flaws into the software systems. There are many reasons for this, one being that software developers do not know what secure code looks like. Information security is usually also at the end of the priority list of software developers when developing software.

Not even educated software engineers know what secure code looks like or how to best take security into account when developing new software. This is due to the fact that information security is largely missing from computer science curriculums, and it's possible to graduate with a degree in computer science without taking a single class on information security. Information security should have a stronger role in computer science education, as it's the graduates who go on to work life and develop the systems we interact with daily.

The goal of this thesis is to understand the current state of the art in academic research on teaching information security, what recommendations and guidelines there exists for information security education. The research will be done as a systematic mapping study. This thesis will attempt to answer the following research questions:

- RQ1: What recommendations and guidelines there exists regarding information security education?
- RQ2: What are the learning goals regarding information security for a computer science graduate?
- RQ3: How can these recommendations and guidelines be taken into account in the computer science education at LUT University?

The research question are answered by conducting the systematic mapping study on teaching information security. The third research question is answered by combining the the answers of RQ1 and RQ2.

This thesis contributes in three ways. Firstly, in this thesis a systematic mapping study on teaching information security is presented. Secondly, the thesis presents guidelines and recommendations for teaching information security at university level. Lastly this thesis presents recommendations on how to take the guidelines and recommendations into account in LUT University's computer science degree program.

The goal of this thesis is only to present guidelines and recommendations for teaching information security. The thesis is not focused on producing information security teaching content for the LUT University's computer science courses. No example course assignments, modules, or teaching packs are presented.

This thesis consists of six sections. The second chapters gives the reader relevant background information, focusing on software vulnerabilities and computer science curriculums. Chapter 3 presents the research process and the conducted systematic mapping study. In the fourth chapter the findings of the systematic mapping study are presented. Chapter 5 presents the results and discussion. Finally, in chapter 6, the conclusions are presented.

2 BACKGROUND

In this chapter, relevant background information is presented. First, the current state of software security and reasons for insecure software is discussed. This is followed by discussion on computer science curriculum guidelines and the role of information security in the curriculum. Lastly, the current state of information security education at LUT University is discussed.

2.1 Software Vulnerabilities

A software vulnerability is ” an instance of a mistake in the specification, development, or configuration of software such that its execution can violate the explicit or implicit security policy” [2]. The definition emphasises that software vulnerabilities can originate from different areas. There exists multiple different kind of software vulnerabilities and mistakes that causes them. Common Weakness Enumeration (CWE) provides a list of most common software weaknesses that can lead to vulnerabilities [3]. Examples of software weaknesses that lead to vulnerabilities are Stack-based Buffer Overflow, Cross-Site Scripting, and Broken Authentication [3]. Vulnerabilities can lead to serious problems for legitimate users for example a buffer overflow can be used by an attacker to execute malicious code [4] allowing the attacker to possibly take over the user’s system or install malware. Cross-Site Sripting can be used to trick the user to install malware or steal sensitive information [5].

The number of sotware vulnerabilities have been rising for the past few years and they show no signs of slowing down in 2018 [6]. It has been thirty years since The Morris Worm introduced the dangers of buffer overflow vulnerability to all internet users at the time [7], but software developers still keep making the same mistakes when developing software [8]. Software vulnerabilities are so common that The Open Web Application Security Project (OWASP) released a document called ”OWASP Top 10” that lists the ten most common software vulnerabilities found in web applications [9]. The OWASP Top 10 document provides information how to test one’s software for these vulnerabilities and how to prevent them from happening. The top vulnerability on the OWASP Top 10 list is called ”Injection” [9]. Injection vulnerabilities allow attackers to perform their own code on the target system and steal information from the target system [10]. The National

Vulnerability Database (NVD) is vulnerability database operated by the U.S government that keep track of all known vulnerabilities in different software [8]. A query for ”sql injection” - a type of an injection vulnerability - provided over 450 results for the year 2018 alone.

Reasons for developers writing vulnerable code are manyfold. D. S. Oliveira et al. Noted that even experienced developers write insecure code [11]. One reason the authors state for vulnerabilities in code is the use of Application Programming Interfaces (APIs) that the developers misuse or misunderstand and thus introduce vulnerabilities in the software [11]. The web is full of forums and blogposts in the form of tutorials on how to develop software. Unruh et. al. Noted that many of the tutorials are flawed and contain bad code examples with software vulnerabilities [12]. The authors noted that many of these bad code examples find their way into real-world code in open source projects. A survey’s results show that security is the last thing on the priority list of sofware developers, it’s more important to ship the features on time [13]. Developers do not know what secure code looks like and lack the skills to spot more advanced security issues [14]. Some of the knowledge gap in secure software is the fact that information security as a topic is not a high priority in computer science university education, computer science students from the top 10 U.S universities can graduate without taking a single course in information security [15].

2.2 Data breaches and EU GDPR

The International Organization for Standards (ISO) and International electrotechnical Commissions (IEC) defines data breach in ISO/IEC 27040:2015 standard as ”compromise of security that leads to the accidental or unlawful destruction, loss, alteration, unauthorized disclosure of, or access to protected data transmitted, stored or otherwise processed” [16]. Trend Micro defines data breach as ”an incident where information is stolen or taken from a system without the knowledge or authorization of the system’s owner” [17]. The stolen data ranges from proprietary trade secrets to confidential information such as credit card data or personally identifiable information such as names, addresses, or social security numbers or persons [17]. According to Trend Micro data

breaches between 2005 and 2015 personally identifiable information was the most stolen type of data, financial data coming in second [17].

Data breaches keep happening and they have been steadily occurring for the past decade [18]. Between April 2018 and June 2018 765 million people were affected by data breaches and the financial losses being in the tens of millions of U.S dollars [19]. Data breaches have negative effects for both the company that got breached as well as the persons whose data was possibly stolen. Companies are affected by diminished reputation, lost customer trust, and reduced revenue [20]. People affected by the breach of personally identifiable information face a life long threat of identity theft and fraud [21]. Sometimes data breaches put people in discomfort in other ways as the Ashley Madison data breach possibly exposed cheating spouses in relationships [22].

There are many ways how data breaches happen: the companies get hacked, malware gets installed onto their systems that steals the information, laptop or mobile phones gets lost, insiders intentionally leak information [17]. Some of the biggest data breaches of 2018 were caused by vulnerabilities in the software systems [23]. In 2017 62% of data breaches featured hacking according to the Verizon 2017 Data Breach Investigation report [24]. In web application based breaches during 2017 an SQL injection was how the data breach happened in 23 of the breaches [24]. According to the Verizon 2015 Data Breach Investigation report 99% of reported vulnerabilities were used a year after the vulnerabilities had been disclosed [24].

Sometimes companies will try to hide the fact that a data breach has happened. Google kept quiet for several months after learning that their Google+ service had vulnerability that exposed personal information of users [25]. Uber decided to hide the fact their services had been breached by paying the attackers in hope that they too will stay quiet about the breach and delete the data [26]. To discourage this sort of behaviour The European Union (EU) developed the General Data Protection Regulation which regulates the actions companies must take when handling personal information of EU citizens [27]. The regulations state that in case of data breach the company must inform about the data breach and prove to be GDPR compliant or face fines of 4 % of annual global turnover or 20 million euros – whichever is higher [27]. The regulation define principles for processing

personal data of EU citizens. These principles affect the software systems that processes the data – the software systems need to be GDPR compliant [28]. Ringmann et al. Identified 74 generic technical requirements based on the GDPR that can be applied to software products that process and handle personal data [28]. Article 5 (1) (f) of the GDPR states that personal ”personal data shall be processed in a manner that ensures appropriate security of the personal data, including protection against unauthorized or unlawful processing and against accidental loss, destruction or damage, using appropriate technical or organisational measures...” [27]. Software developed to be used in the EU or that handles data of EU citizens must take GDPR into account.

The GDPR came in to force on May 25 2018 and by the end of the year there already existed cases where GDPR fines had been issued. A hospital in Portugal was issued a fine of 400 000 € for failure to implement proper access controls [29]. A chat platform in Germany was issued a fine of 20 000 € after it suffered a data breach and the investigations revealed that the user credentials were store in plain text [29] – which is a violation against Article 32 (1) (a) obligation to pseudonymise and encrypt personal data [27].

2.3 Why teach information security and ACM IEEE guidelines

Software vulnerabilities are the root cause for some of the data breaches. The data breaches cause real trouble for businesses and persons. Businesses lose customer trust and thus have reduces revenue when people face fraud and identity thefts. Insecure software systems cause real problems for the society. The rise of vulnerabilities, data breaches, and the fact that even experienced developers write vulnerable code, show a clear need to educate future and current software developers about information security and the importance of it. Software businesses producing services used by EU citizens must make sure their services are GDPR compliant, and this requires the business to have people who have the required information security know-how. It could even be argued that everybody using a computer should be educated about information security, as in 2017 81 % of hacking-related data breaches utilised weak or stolen passwords, 43 % of attacks used some form of social-engineering, over half of the breaches included the use of malware, and 66 % of the

malware was installed by malicious email attachments, according to the Verizon 2017 Data Breach Investigation Report [24].

The Joint Task Force on Computing Curricula Association for Computing Machinery (ACM) IEEE Computer Society provide guidelines for undergraduate degree programs in computer science [30]. They provide the guidelines in form of a report that lists the most essential topics that should be included in a computer science curriculum. The most recent version of the report is Computer Science Curricula 2013 (CS2013) published in December 2013 [30]. The Joint Task Force recognizes the importance of information security and have included information security topics into the newest recommendations: ”In CS2013, the Information Assurance and Security KA is added to the Body Of Knowledge in recognition of the world’s reliance on information technology and its critical role in computer science education” [30].

2.3.1 Topics in an undergraduate computer science curriculum

The Computer Science Curricula 2013 gives a list of recommended topics in an undergraduate computer science curriculum. These topics are organized into 18 Knowledge Areas (KA) which are [30]:

- AL – Algorithms and Complexity
- AR – Architecture and Organization
- CN – Computational Science
- DS – Discrete Structures
- GV – Graphics and Visualization
- HCI – Human-Computer Interaction
- IAS – Information Assurance and Security
- IM – Information Management
- IS – Intelligent Systems
- NC – Networking and Communications
- OS – Operating Systems
- PBD – Platform-based Development
- PD – Parallel and Distributed Computing
- PL – Programming Languages

- SDF – Software Development Fundamentals
- SE – Software Engineering
- SF – Systems Fundamentals
- SP – Social Issues and Professional Practice

These Knowledge Areas are further divided into smaller categories called Knowledge Units (KU) [30]. The Knowledge Unit contains the topics and the corresponding learning outcomes for the unit. The topics are further labeled as ”Core Tier-1”, ”Core Tier-2”, or ”Elective” indicating the importance of the topic in the curriculum [30]. The report states that

- All topics from the Tier-1 core should be included in a curriculum, and the topics should be covered by all students.
- All or almost all Tier-2 core topics should be included, and all students should encounter most of Tier-2 core topics.
    - 90-100% of Tier-2 core topics should be included, 80 % being a minimum.
- The Core topics alone are not sufficient, thus a majority of the electives should be covered as well.

Each learning outcome has an associated level of mastery [30]:

- Familiarity: Understanding what a concept is and what it means.
- Usage: The ability to apply the concept in a concrete manner.
- Assessment: The ability to view the concept from different viewpoints and justify the chosen method for solving a problem.

For each Knowledge Area is given the needed amount of lecture-hours. Lecture-hours indicate ”the time required to present the material in a traditional lecture-oriented format” [30]. This excludes all studying activities outside lectures such as labs and self-study, however it is expected for the students to spend a large amount of additional time outside of lectures further studying the material presented in the lectures [30]. The core hours for each Knowledge Area can be seen in Figure 1 below:

Knowledge Area	CS2013	
	Tier1	Tier2
AL-Algorithms and Complexity	19	9
AR-Architecture and Organization	0	16
CN-Computational Science	1	0
DS-Discrete Structures	37	4
GV-Graphics and Visualization	2	1
HCI-Human-Computer Interaction	4	4
IAS-Information Assurance and Security	3	6
IM-Information Management	1	9
IS-Intelligent Systems	0	10
NC-Networking and Communication	3	7
OS-Operating Systems	4	11
PBD-Platform-based Development	0	0
PD-Parallel and Distributed Computing	5	10
PL-Programming Languages	8	20
SDF-Software Development Fundamentals	43	0
SE-Software Engineering	6	22
SF-Systems Fundamentals	18	9
SP-Social Issues and Professional Practice	11	5
Total Core Hours	**165**	**143**

Figure 1. The Coure Hours for each Knowledge Area.*[30]*

2.3.2 Information security in CS curriculum

Information Assurance and Security (IAS) Knowledge Area is the KA regarding information security in the CS2013 report. The Knowledge Areas in general are interconnected and can be complementary to each other, but IAS is special in the sense that it is distributed across many Knowledge Areas [30] as can be seen from **Figure 2**. Information Assurance and Security Knowledge Area itself has three (3) Core Tier-1 hours and six (6) Core Tier-2 hours. The report states that "Topics germane only to IAS are presented in depth in this KA, whereas other topics are noted and cross referenced to the

KAs that contain them."[30] Respectively IAS has 32 Core-Tier1 hours and 31.5 Core-Tier2 hours in other Knowledge Areas [30].

IAS. Information Assurance and Security "Core" and Distributed

	Core-Tier1 hours	Core-Tier2 hours	Elective Topics
IAS	3	6	Y
IAS distributed in other KA's	32	31.5	Y

Figure 2 IAS core hours distribution [30].

Altogether IAS consists of 35 Core-Tier1 hours and 36.5 Core-Tier2 hours totalling in 63.5 hours. The distribution of lecture-hours among the IAS Knowledge Units can ben seen from **Table 1** below:

Table 1. Distribution of lecture hours in IAS **[1].**

IAS. Information Assurance and Security (3 Core-Tier1 hours, 6 Core-Tier2 hours)

	Core-Tier1 hours	Core-Tier2 hours	Includes Electives
IAS/Foundational Concepts in Security	1		N
IAS/Principles of Secure Design	1	1	N
IAS/Defensive Programming	1	1	Y
IAS/Threats and Attacks		1	N
IAS/Network Security		2	Y
IAS/Cryptography		1	N
IAS/Web Security			Y
IAS/Platform Security			Y
IAS/Security Policy and Governance			Y
IAS/Digital Forensics			Y
IAS/Secure Software Engineering			Y

s

Distribution of IAS lecture-hours in other Knowledge Units can be seen from the **Table 2** below.

Table 2 IAS distribution in other KAs **[30].**

Knowledge Area and Topic	Core-Tier1 hours	Core-Tier2 hours	Includes electives
AR/Assembly Level Machine Organization		1	
AR/Memory System Organization and Architecture		0.5	
AR/Multiprocessing and Alternative Architectures			Y
HCI/Foundations	1		
HCI/Human Factors and Security			Y
IM/Information Management Concepts	0.5	0.5	
IM/Transaction Processing			Y
IM/Distributed Databases			Y
IS/Reasoning Under Uncertainty			Y
NC/Introduction	1		
NC/Networked Applications	0.5		
NC/Reliable Data Delivery		1.5	
NC/Routing and Forwarding		1	
NC/Local Area Networks		1	
NC/Resource Allocation		0.5	
NC/Mobility		1	
OS/Overview of OS	2		
OS/OS Principles	1		
OS/Concurrency		1.5	

Table 2 continued.

OS/Scheduling and Dispatch		2	
OS/Memory Management		2	
OS/Security and Protection			
OS/Virtual Machines			Y
OS/Device Management			Y
OS/File Systems			Y
OS/Real Time and Embedded Systems			Y
OS/Fault Tolerance			Y
OS/System Performance Evaluation			Y
PBD/Web Platforms			Y
PBD/Mobile Platforms			Y
PBD/Industrial Platforms			Y
PD/Parallelism Fundamentals	1		
PDParallel Decomposition	0.5		
PD/Communication and Coordination	1	1	Y
PD/Parallel Architecture	0.5		Y
PD/Distributed Systems			Y
PD/Cloud Computing			Y
PL/Object-Oriented Programming	1	3	
PL/Functional Programming	1		

Table 2 continued.

PL/Basic Type Systems	0.5	2	
PL/Language Translation and Execution		1	
PL/Runtime Systems			Y
PL/Static Analysis			Y
PL/Concurrency and Parallelism			Y
PL/Type Systems			Y
SDF/Fundamental Programming Concepts	1		
SDF/Development Methods	8		
SE/Software Processes	1		
SE/Software Project Management		1	Y
SE/Tools and Environments		1	
SE/Software Construction		2	Y
SE/Software Verification and Validation		1	Y
SE/Software Evolution		1.5	
SE/Software Reliability	1		
SF/Cross-Layer Communications	3		
SF/Parallelism	1		
SF/Resource Allocation and Scheduling	0.5		
SF/Virtualization and Isolation		1	
SF/Reliability through Redundancy		2	
SP/Social Context	1		
SP/Analytical Tools	1		

Table 2 continued.

SP/Professional Ethics	1	0.5	
SP/Intellectual Property	2		Y
SP/Analytical Tools			
SP/Privacy and Civil Liberties	0.5		
SP/Security Policies, Laws and Computer Crimes			Y

The Information Assurance and Security Knowledge Area contains many topics with only nine (9) lecture-hours to cover them [30]. However this is balanced in the way that level of mastery in the learning outcomes is "familiarity" for the most topics [30]. The report states that "The broad application of the IAS KA concepts (63.5 hours) across all other KAs provides the depth of coverage and master for an undergraduate computer science student."[30]

2.4 Current situation at LUT University

Lappeenranta-Lahti University of Technology LUT provides a Bachelor's Programme in Software Engineering that is taught in Finnish. The learning outcomes on completion of the program of the programme are as follows [31]:

1. Apply software engineering theory, principles, tools and processes, as well as the theory and principles of computer science and mathematics, to development of complex, scalable software systems
2. Demonstrate software engineering application domain knowledge and principles of selecting and the use of software matrices
3. Understand the dynamics of how teams develop and function, productively participate on software project with heterogeneous teams
4. Interact professionally with colleagues or clients and overcome challenges that arise from geographic distance, cultural differences, and multiple languages in the context of computing and software engineering

5. Communicate effectively both verbally and in writing, produce documents, and work as a part of a project team using both the domestic languages as well as English
6. Recognize the need for, and engage in, lifelong learning
7. Describe, design and solve problems by programming and using software engineering techniques and experimentation
8. Apply technical skills in different application domains taking into account technical, social, an economical constraints
9. Elicit, analyze and specify software requirements through a productive working relationship with project stakeholders
10. Apply appropriate codes of ethics and professional conduct to the solution of software engineering problems
11. Understand IT related business, entrepreneurship and innovation models.

The learning outcomes largely follow the characteristics of computer science graduates as defined by The CS2013 report [30] with the exception of not explicitly stating the recognition of information security as a core characteristic of graduates.

Information security is largely missing from the Bachelor's Programme in Software Engineering at LUT University. Currently as it stands, much like the top 10 U.S universities [15], it is also possible to graduate from LUT University with a Bachelor's Degree in Software Engineering without taking a single course on information security as the University does not provide a single course on information security [31]. Only four out of 31 mandatory courses in the Bachelor's Programme have mentioned information security in the topics of the course [31]. This means that much of the information security content of the other courses largely lie on the shoulders of the lecturer. Then there exists the problem of what content and topics to include, as the lecturer might not have the subject expertise in information security. There clearly exists a need to uniform the information security education in the Software Engineering degree programme at LUT.

3 RESEARCH PROCESS

In this chapter the used research method and the research process is presented. First the research method and motivations for choosing it are presented. This is followed by presenting the conducted research process.

3.1 Systematic Mapping Study

The aim of this research is to understand the current state of the art in academic research on teaching information security, what recommendations and guidelines there exists for information security education. For this purpose the systematic mapping study was selected.

Systematic mapping study is a secondary study method that is used to provide an overview of a research area [32]. It provides structure on previous research by categorizing the research reports and results, presenting a visual summary of them [32]. With it the research can be carried out in a repeatable manner as each step of the process can be systematically followed. The systematic mapping process suggested by Petersen et al. is presented in **Figure 3** below:

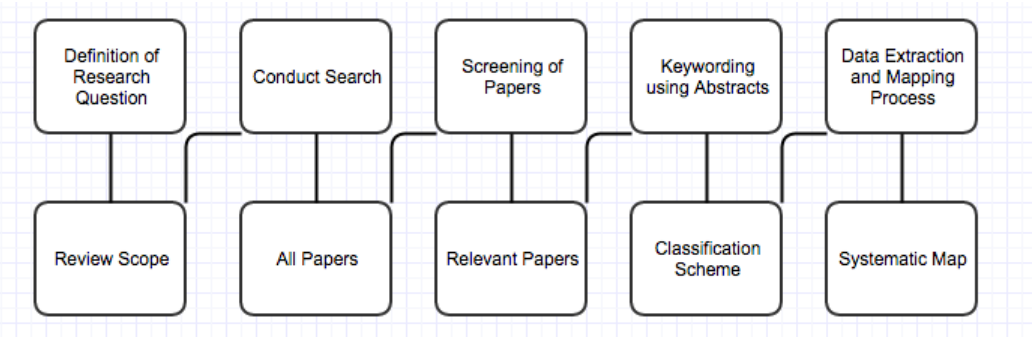

Figure 3 The Systematic Mapping Process *[32]*

The process consists of five steps, shown on the top row in **Figure 3**, and the outcomes of the corresponding steps, shown on the bottom row of **Figure 3**. The process begins with the definition of the research questions. The next step is to carry out the search from the

selected journals, databases or conferences with the selected keywords. The articles are then screened for inclusion and exclusion based on the inclusion and exclusion criteria resulting in the relevant papers. The relevant papers are then key worded using the abstracts providing us the classification scheme. The last step is data extraction and mapping process in which the articles are sorted into the scheme, resulting in the Systematic Map [32].

3.2 Conducting the search

Instead of selecting specific journals or conferences the following scientific databases were used to cover a wider range publications: Web of Science, ACM Digital Library, and IEEEXplore. These databases provide large set of journals and conferences in computer science, software engineering, and information systems. Also all of them offer advanced search functionality that allow for more fine grained queries, such as searching only in the title, metadata or both.

Selection criteria is used to exclude studies that are irrelevant in answering the research questions [32]. As the goal in this research was to find out the state of the art on information security education in university level, articles regarding teaching information security at university level were considered relevant. Papers explicitly mentioning teaching information security or teaching cyber security in title or abstract were included. If the articles were not about teaching information security or were about information security education outside university level, they were excluded. The selection criteria was applied systematically by first excluding irrelevant papers by title and abstract. Finally all papers were read in full to determine final inclusion or exclusion. The following inclusion and exclusion criteria were used:

The inclusion criteria for the articles:

- The title or abstract of the article explicitly mentions teaching information security
- The title or abstract of the article explicitly mentions teaching cyber security
- The abstract mentions teaching an information security topic at university level

The exclusion criteria for the articles:

- The article was about information security education but not at university level

- The article was not fully available
- The article was not written in English
- The article was about information security but not about teaching information security at university level

3.2.1 Pilot search

In Systematic Mapping Studies the search strings are formed on the basis of the research questions [32]. The research questions for the research are the following:

- RQ1: What recommendations and guidelines there exists regarding information security education?
- RQ2: What are the learning goals regarding information security for a computer science graduate?
- RQ3: How can these recommendations and guidelines be taken into account in the computer science education at LUT University?

The mapping study will be used to answer the first two research questions. The third research question will be answered by analysing the results of the systematic mapping study, and it thus will not affect the search strings.

First a set of pilot searches from two different databases were conducted in order to test the feasibility of the different search strings. The feasibility was evaluated by the number of search results the string produced, and randomly selecting five results from the search and evaluating the inclusion criteria on them.

The first pilot search was conducted with ”teaching AND information AND security” and ”information AND security AND education” as the search strings. The pilot searches resulted in a large number of results, as can be seen from **Table 3**. However after taking a closer look at the titles and abstracts of the results it was clear that the search strings needed to be altered as results contained irrelevant articles, such as ”Study on Security and Stability Of Prison Management” and ”Assessing the efficiency of public education and pensions”. For the second pilot search the following search strings were used:

- (teaching AND ”information security”)
- (teaching AND ”cyber security”)

The second pilot search resulted in much reasonable amount of search results as shown in **Table 4.**

Table 3 Results from first pilot search

Source	**Search string**	**# Results**
ACM DL	Teaching information security	231 490
IEEEXplore	Teaching information security	563
ACM DL	Information security education	252 056
IEEEXplore	Information security education	3801

Table 4 Results from second pilot search

Database	**Search string**	**# of results**	**# out of / 5 random**
IEEEXplore	(teaching AND ”information security”)	191	3/5
IEEEXplore	(teaching AND ”cybersecurity”)	59	4/5
ACM DL	(teaching AND ”information security”)	123	3/5
ACM DL	(teaching AND ”cybersecurity”)	70	4/5

For each search round in the second pilot search five articles were randomly chosen from the results. They were evaluated against the selection criteria for the validity of the search strings. As can be seen from **Table 4** in total of 14 out of 20 were considered as relevant. This suggested that the keywords produced sensible results.

3.2.2 Performing the search

After seeing the results of the pilot searches the search strings from the second pilot search was decided to be the final search strings. However the time period was set from 2010 to 2018 so that the results presented more the recent knowledge and advancements that better

recognize the current problems and solutions regarding information security education. After doing the searches, applying the selection criteria, and filtering out duplicates the search resulted in total of 86 unique articles. The results of the actual searches can be seen in **Table 5**.

Table 5 Results from actual searches

Database / Search string	(teaching AND ”information security”)	(teaching AND ”cybersecurity”)	Accepted / Total
ACM DL	28 / 60	20 / 65	48 / 125
IEEE Xplore	26 / 107	11 / 58	37 / 165
Web Of Science	13 / 179	8 / 52	21 / 231
Total	67 / 346	39 / 175	106 / 521
Total after removing duplicates			86 / 521

4 FINDINGS

The categorization of the articles was done by reading through the each of the accepted articles and evaluating how they discussed topics related to the research questions. From each article the title, publication year, and the most essential topic of interest was recorded. 21 articles were rejected on closer reading as they satisfied exclusion criterias: paper was not fully available or the article was about teaching information security but not at university level. This reduced the total number of accepted articles down to 65 unique articles. Four main categories were identified from the papers: *Laboratory environments, Teaching specific topics, How to teach information security,* and *Course descriptions and teaching experiences*. Articles describing laboratories that are used to teach information security are categorized under ”Laboratory environments”. ”Teaching specific topics” deal with articles that discuss teaching one certain or a set or particular topics regarding information security. Articles describing teaching practices and approaches were categorized under ”How to teach information security”. If the article presented a course, curriculum, or teaching experiences it was categorized under ”Course descriptions and teaching experiences”. The selected set of articles are presented in Appendix 1.

Ten of the articles addressed *laboratory environments* that were used to teach information security topics. Information security educations requires hands-on exercises, both defensive and offensive in nature. To incorporate the hands-on aspect into courses a laboratory settings is needed to safely let student experiment with the different cybersecurity attacks. A physical laboratory setting can be implemented to let students learn in a safe environment. However the physical laboratory settings were considered troublesome as they need constant care taking and management, the equipment costs, they may pose a threat to the universities’ campus networks. Virtual Machines Laboratories are proposed as a solution as they remediate many of the problems physical laboratories have.

21 of articles focused on *teaching specific topics* in information security. The topics range from network attacks to forensics. The topics were largely focused on teaching offensive information security skills. Articles that focused more on the offensive side of information security deal with teaching different variations of certain attacks or a mix of different

attacks. A number of articles address forensics or identifying whether has an attack occurred.

24 articles address *how to teach information security*. These articles discuss different learning and pedagogical approaches used in information security education, and frameworks for teaching different information security topics. Ten publication presented *Course descriptions and teaching experiences.* These articles present course curriculums, lab assignments for the courses, and discuss experiences the authors had with teaching the courses.

Figure 4 shows the distribution of of articles in each category. As can be seen that most of the done research is on *How to Teach Information Security* and *Teaching Specific Topics*.

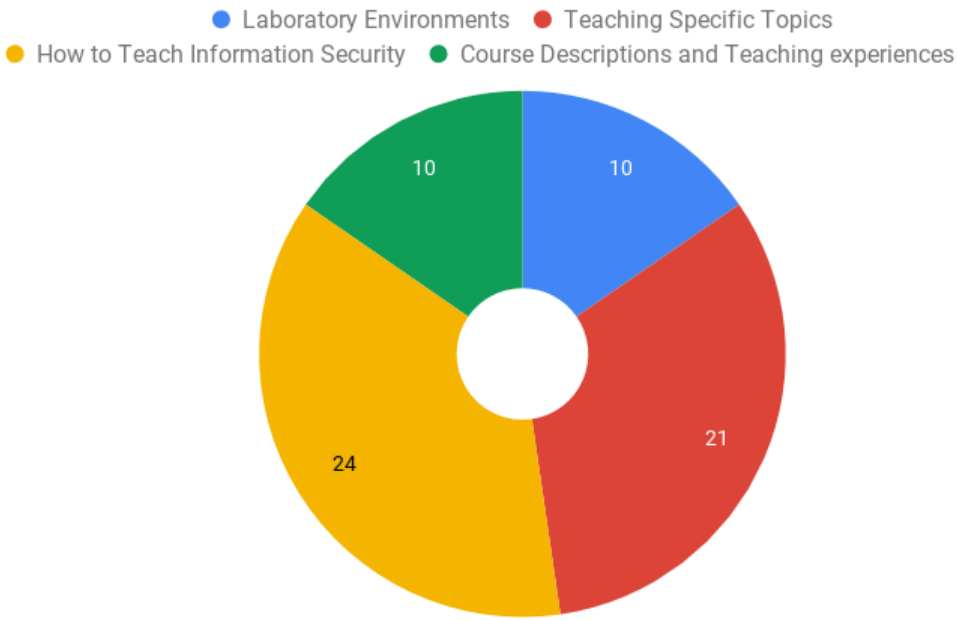

Figure 4 Pie chart on number of articles in each category

Figure 5 shows the distribution of the articles in each category by the publication year. It can be seen that after 2010 publications in *Laboratory Environments* and *Teaching Specific Topics* have emerged. It can also be seen that publication numbers per year is relatively stable, except in the year 2016. The small number of articles in the year 2019 is explained by the fact that the research was done in the beginning of the year 2019.

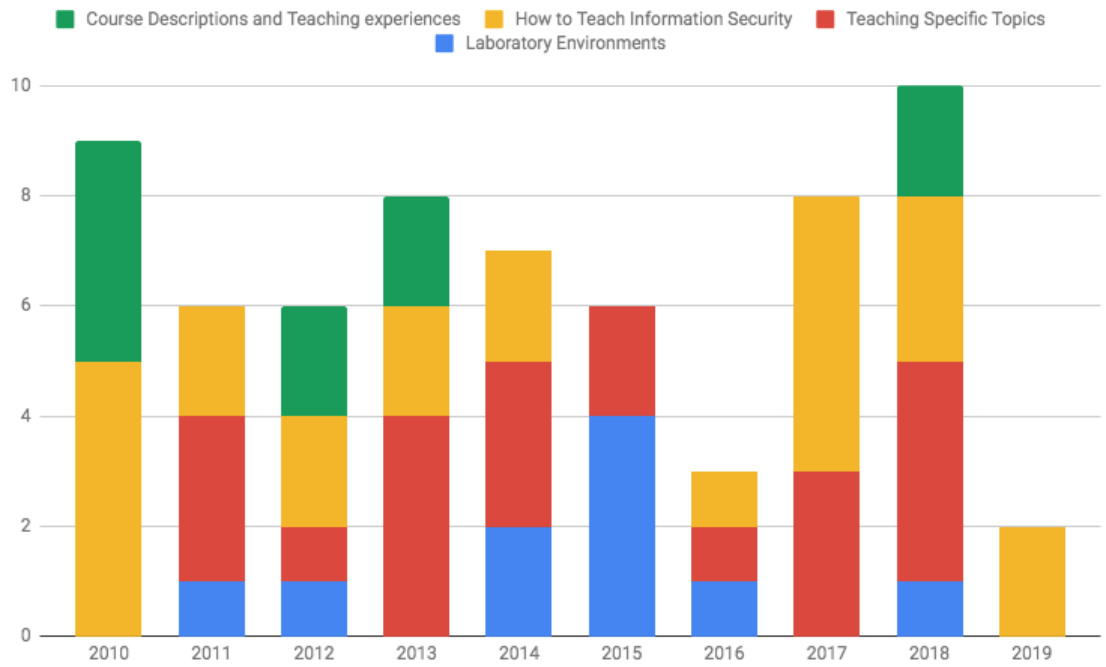

Figure 5 Articles in each category by publication year

Systematic map of teaching information security is presented in **Table 6**. The categories and articles for each category is presented in the table. The next section walks through the main findings from the articles.

Table 6 a Systematic Map

Laboratory Environments	Teaching Specific Topics	How to Teach Information Security	Course Descriptions and Teaching Experiences
(Peltsverger and Zhang, 2014) (Willems et al., 2011) (Salah, 2014) (Eliot et al., 2018) (Kaabi et al., 2016) (Timchenko and Starobinski, 2015) (Mirkovic and Benzel, 2012) (Tunc et al., 2015) (Salah et al., 2015) (Weiss et al., 2015)	(Jillepalli et al., 2018) (Weiss and Mache, 2011) (Trabelsi, 2014) (Ledford et al., 2016) (Damon et al., 2012) (Trabelsi, 2011) (Trabelsi and Ibrahim, 2013) (Trabelsi and Alketbi, 2013) (Trabelsi and Saleous, 2018) (Dimkov et al., 2011, 2011) (Katsadouros et al., 2017) (Nguyen and Irvine, 2018) (Pan et al., 2015) (Nordhaug et al., 2014) (Simpkins et al., 2015) (Hamdan, 2017) (Trabelsi and Mustafa, 2014) (Wen et al., 2017) (Zhu et al., 2013) (Haywood et al., 2013) (Landry, 2010) (Figueroa et al., 2018)	(Konak et al., 2013) (Konak et al., 2014) (Yuan et al., 2017) (Bhatt et al., 2018) (Luburić et al., 2019) (Chatmon et al., 2010) (Yuan et al., 2010) (Vaidya et al., 2014) (Ktoridou and Dionysiou, 2011) (Cai and Arney, 2017) (Ahmed and Roussev, 2018) (Alhamdani, 2016) (Peltsverger and Karam, 2010) (McManus, 2018) (Bratus et al., 2010) (Mink and Greifeneder, 2010) (Wilson, 2017) (Conti and Caroland, 2011) (Hamman et al., 2017) (Vorobeychik et al., 2013) (Joshi et al., 2012) (Voorhees et al., 2017) (McDonald and Andel, 2012) (González-Manzano and de Fuentes, 2019)	Uskov, 2013a) (Meghanathan et al., 2012) (Uskov, 2013b) (Wang et al., 2018) (Švábenský et al., 2018) (Aman et al., 2010) (Raj and Savacool, 2010) (Jovanovic and Mirzoev, 2010) (Wei and Sun, 2012) (Sun, 2010)

4.1 Laboratory environments

Ten on the articles address virtual machine laboratories that were developed to aid in teaching hands-on skills for students. Teaching students using practical hands-on exercises is considered an essential part of information security education [33]. The natural teaching environment for teaching the hands-on skills normally is a campus laboratory [34], however physical laboratories were noticed to have problems of their own. Salah [35] listed the cost of hardware equipment, the need for the laboratory environment to be managed, and instructors time is spent troubleshooting the lab equipment rather than teaching the students, as problems of traditional physical laboratory environments. Often there is also a need to give students full access to the hardware and software which is troublesome in physical laboratory settings, as it imposes security risks to the university's campus network [36]. To address the issues of physical laboratories the articles in this category propose the use of virtualized laboratories.

Three of the ten articles proposed the use of local virtualization laboratories, environments that can be locally installed into the computers of students or into the physical laboratories' computers. Peltsverger and Zhang [33] present a virtualization laboratory they built using NetKit which makes creating virtualized networks easy for the students. Khaabi etal. [37] present local virtualization lab specifically designed to teach students DOS attacks in a safe environment as the attacks can not escape from the virtual environment out to the real world. Many of the proposed virtualization approaches were considered often increased in complexity [38] and thus the authors propose a very simple virtual laboratory environment with a set of hands-on exercises that anyone can run on their computer.

[34], [36], and [39] present centralized virtualization solutions. In centralized virtualization all virtual machines run on the same physical servers that handle the virtualization. In [36] the authors present the physical laboratory they built to be used as a centralized virtualization server off-campus. The authors also describe honeypots that were developed and installed into the laboratory for the students to be able to disseminate the attacks instead of just doing them. Willems et al [34] present a distributed laboratory architecture called Tele-Lab. Tele-lab is accessed by students using a web browser where all the

training takes place. The laboratory is distributed among two universities in different countries and is scalable as the universities can lend computing resources one another when required. [39] discuss DeterLab, a centralized solution that consist of over 400 computing nodes and special hardware. DeterLab can be used by instructors by applying for a DeterLab project, and it's already used by 47 institutions worldwide. The key difference in DeterLab that is is also used as a research platform instead of just an education platform.

Four of the articles discussed cloud based virtualization laboratories. In cloud based virtualization the physical equipment management is outsourced to a third party and the virtualization is provided as a cloud service. In [40] ClaaS (Cybersecurity Lab as a Service) is presented. The authors state that advantages of using a cloud based solution is that there are no physical or local virtual labs to setup. In addition the laboratory can be accessed from any device using just a web browser. In [35] and [41] the authors discuss the use of AWS (Amazon Web Services) as a virtual lab environment and the advantages of the cloud compared to physical labs. Main advantages were: pre-made images that have all the necessary software installed so students do not have to install anything themselves, easy creation of virtual machine instances, students can learn in a safe environment in the cloud separated from the rest of the internet, and instructor can easily log into students virtual machines for grading the assignments [35]. However the cloud is not perfect as some exercises can not be performed in the cloud such as Wireless Network attacks or smartphone forensics [41]. Weiss et al [42] present EDURange framework, a cloud service designed to specifically teach analysis skills. By using the cloud the exercises can be accessed from anywhere and by anyone using a browser. The authors state that EDURange is not meant to replace other lab environments but provide a platform with easily accessible exercises that teach analysis skills.

4.2 Teaching specific topics

22 of the articles address teaching specific topics in information security. These topics range from network attacks to forensics. Teaching information security using hands-on exercises is considered a best-practice in information security education [43] as students learn best with hands-on experience [44]. The articles in this category focus on describing different hands-on exercises that are used for teaching information security concepts.

A large number of these articles focus on teaching offensive security skills using hands-on exercises. Teaching these offensive security skills is seen as a necessary component in information security education as it produces better security professionals compared to teaching defensive techniques alone [45]. [46], [47], [48], [49], and [50] describe hands-on exercises for teaching Denial Of Service (DOS) attacks. [47], [48], [49] and [50] focus on teaching different variations of Denial of Service attacks. In [46] the authors describe a DOS attack exercises that was used teach scientific experimentation. DOS attack is seen as an important topic as it's widely used by attackers and understating the attacks is they key for defending against them: "One cannot build or architect defences for attacks that one does not truly understand" [48]. [45] and [51] focused on network eavesdropping attacks conducted in local area networks. In [52] the authors present a physical penetration test assignment they used in a computer security course. The authors argue that the social and physical security aspects are often neglected in computer security courses, and this gives the students an unrealistic view of information security if the focus is only on the technical side. In [44] the authors argue that security education should be integrated into all core computer science classes and present a wide range of hands-on offensive security exercises that can be incorporated into different core computer science courses. In [53] the authors present a scavenger hunt game that is used to teach wireless network security and attacks. In [43] the authors present eight network security tutorials that anyone can use to teach network security.

Out of the 22 articles eleven of them focus more on teaching defensive skills using hands-on exercises. [54], [55], and [56] focus on teaching digital forensics. [55] and [56] both present frameworks which are used to teach forensics utilizing game-based learning. In [54] the authors describe the hands-on teaching material they used to teach industrial

network forensics. [57] describe a course module on web tracking and web privacy issues. In [58] the authors describe teaching data analytics for identifying potential intrusion in a web server. [59] describes Edu-Firewall, a web-based firewall simulator tool. With it firewall configurations can be taught to students without buying expensive hardware or software firewalls. [60] presents What.Hack game that teaches students to identify phishing attacks which is one of the most common social engineering attacks today. Two articles focused on secure programming practices. In [61] the authors argue that many of todays information security vulnerabilities stem from software flaws and this root cause is often neglected in traditional information security curricula. To help students learn secure programming practices to avoid software vulnerabilities the authors present a plugin for integrated development environment that teaches secure programming practise. [62] argue that it's important for developers to know the language features that support secure development and focused on teaching such structures using Java programming language as an example. In [63] the authors describe a hands-on case study assignment that was used in an information security management course to teach risk assessment. [64] presents a detailed introduction to RFID security analysis in the context of IoT utilizing problem based learning approach.

4.3 How to teach information security

A fair number of the articles address how information security can be taught. These articles discuss about frameworks for teaching information security topics, different pedagogy and learning approaches.

Five articles present frameworks for teaching information security topics. Hands-on laboratory assignments are considered a best practice in information security education. In [65] and [66] the authors present a framework for designing hands-on laboratory assignments for information security courses. When designing hands-on laboratory assignments using open-ended questions versus following specific instructions is considered equally effective [67]. [68] argue that most offensive hands-on laboratory exercises focus only on user level attacks and presents a framework for teaching kernel level security using virtual machines. Security design analysis is considered an important step in software development but it is hard to teach and learn. In [69] the authors present a framework for teaching security design analysis.

A fair number of the articles discuss different learning approaches and pedagogy in information security education. Active learning approaches (learning by doing) are considered good practice as they result in deeper understanding of concepts [70]. There are many kinds of active learning techniques such as hands-on virtual machine assignments, case-studies, group discussions, and peer-instruction [70]. After incorporating hands-on exercises in cryptography courses the authors measured clear improvements regarding student's understanding of the material. In [71] the authors present how they utilised case studies for teaching physical security and security policy. Vaidya et al. [72] present how case studies and game-based learning can be combined to teach privacy. Case-based learning can also be used to integrate information security into other computer science courses [73]. Information security is often taught bottom-up going through a list of topics taught in isolated context from each other [74]. In [74] the authors argues that information security should be taught top-down and case-driven by using real-world security breaches as an example, disseminating them from the beginning to the end. This way the it is clear how the topics relate to each other. [75] presents a methodology for developing peer

instruction questions and discuss teaching experience using them in cyber security courses. In [76] the authors present a teaching model for teaching cryptography that is based on Design Thinking approach. Most students keep programming the way they first learn to program in university courses [77]. Therefore it's important to teach security concepts at the same time as the main course topics using ”security in mind” approach [77]. Information security cannot be bolted on to software as an afterthought [78]. Teaching software development utilizing security by design approach students learn to treat security as a set of design requirements that are addressed throughout the software development lifecycle [78]. [79] argues for teaching the ”Hacker Curriculum”: Students should be taught to develop a cross-layer view of the systems and to understand them by analysing their failure modes. Most computer science courses concentrate thinking on one layer of abstraction at a time and students will continue to think this way through their professional career. Engineering a successful exploit requires understanding the system on many different layers of abstraction and failure modes, and successful defences also require the same cross-layer approach [79].

Teaching information security using the offensive approach, ie. teaching the students how to attack, is considered good practice. In [80] the authors evaluated the offensive approach by teaching the same course first using the offensive approach and then using the defensive approach. Compared to the defensive approach teaching security in the offensive approach resulted in better understanding of information security and the students were more more motivated. Students who had taken an offensive security course developed more secure software when compared to students with defensive security training [81]. Teaching offensive security better prepares students to adapt to new attacks and makes them more able to protect software systems as they truly understand the attacks better [81].

On top of understanding how the attacks work students should also be taught to think like an attacker as well [79]. Neglecting adversarial mindset in information security education puts the students at a disadvantage against opponents who don't play by the rules [82]. By teaching the students adversarial thinking they are better prepared to reason when, where, and how the attackers will attack [83]. Adversarial thinking can be taught by teaching basic game theory concepts [83]. Conti and Caroland discuss experiences teaching adversarial thinking by making the students cheat in a test [82]. Bratus et al. argue that ”To meet the

challenges of modern computer security practice, students must be able to switch from their conditioning by traditional computer science and software engineering curricula to the attacker's way of thinking" [79].

Capture The Flag (CTF) competitions are seen as a great way to teach students of all skill levels [84]. They help students to internalize and apply security principles that students might only know in theory, and are feasible in teaching students to implement more secure software [84]. When designed properly CTFs can provide realistic scenarios in exploiting and fixing software vulnerabilities in a highly motivating manner [85].

Teaching information security can be problematic when institutions lack the required expertise and the existing faculty is already swamped with courses to teach rendering new courses unfeasible [86]. Information security topics can be injected across the existing computer science curriculum, avoiding the need for new courses and increase in faculty. This way students are encouraged to think information security as an essential part of their skillset [86]. Vorhees et al. discuss their approach in injecting the CS2013 tier 1 and tier 2 IAS outcomes into the existing computer science courses at their university [86]. They organized the 37 tier 1 and tier 2 IAS outcomes from the CS2013 into nine information security topics and mapped those topics to courses found in most common computer science programs. Based on the categorization they formed the following learning goals for information security of a computer science program [86]:

- "1. Be able to articulate foundational concepts in securing networks and systems.
- 2. Be able to apply principles of secure design and defensive programming techniques when developing software.
- 3. Be able to identify security issues associated with common threats and attacks."

They also developed teaching modules for injecting information security content into courses for other computer science instructors to use in their curriculum. McDonal and Andel argue that certain principles of information security research are timeless and foundational [87]. They used the principles to developed learning objectives that can be integrated into computer science curriculum [87].

There are many information security courses provided by online educational platforms that teach a wide variety of topics [88]. When designing an information security Massive

Online Open Course (MOOC) there exists the questions of what to teach and how to teach it. González-Manzano and de Fuentes present recommendations for instructors designing information security MOOCs on what to teach and how to do it [88].

4.4 Course descriptions and teaching experiences

As information security has become an essential part of computer science education [30], universities have begun to develop completely new courses [89], and incorporating information security topics into existing courses [90]. Articles in this category describe the new or altered courses that have been developed to teach information security and meet the requirements of the ACM/IEEE guidelines. These articles focus on providing the course descriptions and teaching experiences.

Uskov reported on two new courses that are part of an effort to make the computer science curriculum of Bradley University to meet the Information Assurance requirements of the CS2013 report [89] [91]. In [89] the author presents a new course on applied cryptography. The article describes the courses's topics and how they map onto the CS2013 curriculum requirements, teaching approach, schedule for the course, lab assignments, and teaching experiences. In [91] a new course on software and web application security is presented. The author presents the course topics and how they map to the requirements of CS2013, and provides a framework he developed for teaching different software vulnerabilities and attacks.

Information security education has many difficulties: it's challenging to choose what topics to teach, the field advances rapidly requiring lots of effort from educators to keep courses up-to-date, industry places a lot of value on certificates, and students favour attack-oriented courses over purely defence oriented ones [92]. Wang et al. [92] presents a course developed to tackle the previous issues. The authors discuss how topics are organized into a security spectrum that maintain a logical flow. To keep the course up-to-date the authors included a module of "emerging topics" that can be used to teach timely issues and new developments. Keeping the students motivation high the course has hands-on labs that provide equal emphasis on offensive and defensive skills, and incorporated key knowledge from entry level certifications into the course curriculum [92].

Švábenský et al. describe two information security courses that are developed to teach students adversarial thinking by making the students create attack and defence educational

games [93]. The authors describe the courses' design, content, assessment methods, and share their teaching experiences. In the authors experience students highly value the "learn by teaching" method of the courses, and other students who played the developed games rated the educational value of the games highly [93].

Aman et al. [94] describe a capstone exercise of an information security course. The authors present the exercise, details of the setup, and teaching experiences. Meghanathan et al. [90] share their experiences from incorporating information security into software engineering capstone course sequence. The authors were motivated to include aspects of information security into the course sequence as security cannot be bolted onto software as an after-thought [90]. As a result the students were more confident in their ability to incorporate security into software development, and the students realized the importance of of security in all phases of software development life cycle [90].

In [95] and [96] the authors share their teaching experiences on courses focusing on data management and information security. Raj and Savacool presents Secure Data Management course, a topic the authors feel is missing from many curriculums [95]. Jovanovic and Mirzoev [96] argue that education has not kept up with the technological advances in storage networks and their security. The authors describe two courses on secure storage infrastructure management in an effort to tackle the issue and share their experiences teaching the courses.

Wei and sun [97] present their experiences in teaching "Network Attacks and Defence" course. In the course students were either attackers attacking a computer system or defenders defending against the attack, providing students hands-on experience in both sides [97]. In the authors experience this kind of hands-on learning engaged the students in self-directed learning when neither the attackers nor defenders wanted to fail in their assignments [97].

In [98] Sun describes experiences in teaching a time-condensed computer security class. The author condensed a 16 week class into a 6 week long class. The author presents the course topics, schedule, and lab assignments. Each component of the original course was

customized to fit the shorter time-condensed [98]. In the authors experience the time-condensed version was as effective as the longer course, and the condensed version enables students to learn computer security basics at a faster pace making the students better prepared for the fast-evolving security threats [98].

5 RESULTS AND DISCUSSION

The goal of this research is to understand the current state of the art in academic research on teaching information security, what recommendations and guidelines there exists for information security education. For this purpose a systematic mapping study was conducted. Four main categories were identified in the mapping study from the articles: *Laboratory environments, Teaching specific topics, How to teach information security,* and *Course descriptions and teaching experiences*. These categories are more thoroughly presented in chapter 4. In the first chapter three research questions were defined. The research questions are: *RQ1: What recommendations and guidelines there exists regarding information security education?, RQ2: What are the learning objectives regarding information security for a computer science graduate?,* and *RQ3: How can the recommendations and guidelines be taken into account in the computer science education at LUT University?* In this chapter the answers to these research questions are presented.

5.1 What recommendations and guidelines there exists regarding information security education?

The Joint Task Force on Computing Curricula Association for Computing Machinery (ACM) IEEE Computer Society provides the CS2013 curriculum guidelines for computer science degrees [30]. The report provides guidelines on what information security topics should be included in computer science curriculum, what are the learning outcomes of the topics, and what is the minimum amount of time that should be spent on the subjects.

Literature provided many recommendations for information security education. Information security education should utilize hands-on exercises as students learn best with hands-on experience. The hands-on exercises should be provided in safe laboratory environment to prevent students from accidentally attacking real world systems. A recommendation is to use virtualized laboratory environments as they provide many benefits over a physical laboratory environment. Teaching information security using the offensive approach is considered good practice as it keeps the students motivated, provides better results compared to defensive education, and makes students more able to protect

software systems as they truly understand the attacks. Students should be taught adversarial mindset, ie. to think like an attacker. The adversarial mindset helps the students to reason when, where, and how the attackers will attack. Information security should be injected across the curriculum as this way students are encouraged to think information security as an essential part of their skillset. This view is mirrored by the CS2013 report as well [30]. By injecting the information security across the curriculum into other courses, information security is thought of in most courses keeping it in the students' minds preventing it from becoming an afterthought.

5.2 What are the learning goals for computer science graduate regarding information security?

The CS2013 report itself does not define any explicit learning goals, but it lists the learning outcomes for each Information Assurance and Security knowledge unit [30]. As presented in chapter 2, the CS2013 report organized computer science Body Of Knowledge into Knowledge Areas. A Knowledge Area corresponds to a topical area of study in computing, Operating Systems and Software Engineering being two examples [30]. Information Assurance and Security (IAS) is the Knowledge Area related to information security. A Knowledge Area is further divided into Knowledge Units. Each Knowledge Unit has a set of topics and learning outcomes. The topics and learning outcomes in a Knowledge Area are organized into three groups: tier 1, tier 2, and elective. All tier 1 topics (and learning outcomes) should be covered in a curriculum, at least 80% of the tier 2 topics, and as many as possible of the electives [30]. There are only three knowledge units in IAS that contain tier 1 learning outcomes and five knowledge units that contain tier 2 learning outcomes. The knowledge units that contain tier 1 learning outcomes are [30]:

- Foundational concepts in security
- Principles of secure design
- Defensive programming.

IAS knowledge units that contain tier 2 learning outcomes are:

- Principles of secure design
- Defensive programming
- Threats and attacks
- Network security

- Cryptography

These can be seen as high level themes computer science graduates should be familiar with. However the CS2013 report states that many of the information security topics are spread across the knowledge areas making information security a cross curriculum theme.

Vorhees et al. defined the following information security learning goals for students of computer science, based on the learning outcomes in the CS2013 report [86]:"

- Be able to articulate foundational concepts in securing networks and systems.
- Be able to apply principles of secure design and defensive programming techniques when developing software.
- Be able to identify security issues associated with common threats and attacks."

The views of the literature largely follow the learning goals defined by Vorhees et al. According to the literature review computer science graduate need to possess an adversarial mindset, they need to be able to think like an attacker. The graduates should know how to attack and how the attacks work to be able to defend against old and new attacks. They should develop a cross-layer view of the systems as many of the vulnerabilities, exploits, and their defences need understanding of the system at many level of abstractions. Graduates should take information security into account in every phase of the software development lifecycle.

When developing software, the ability to think link an attacker helps one identify possible security weaknesses in the software and prevent security bugs. The same goes for knowing how to attack and how the attacks work. For example when a developer knows how to perform SQL injection attacks and how they work the developer can produce software that mitigates such attacks. The adversarial mindset and knowledge can be applied in every phase of software development, making security a "requirement" and not an afterthought. Understanding system's abstractions and the boundaries and relations of those abstractions are vital in understanding the attacks and the ways the systems can be attacked.

Information security is a wide topic and it's unreasonable to think that any computer science curriculum could cover all of it. There are many other things the computer science

curriculum should cover as well. The are a total of 37 learning outcomes for Information Assurance and Security knowledge area in the CS2013 report [30]. The 37 learning outcomes consist of only tier 1 and tier 2 learning outcomes. There are 17 tier 1 learning outcomes and 20 tier 2 learning outcomes. The report states that all of tier 1 should be covered by the curriculum and at least 80% of the tier 2 should be covered [30].

5.3 How can the recommendations and guidelines be taken into account in the computer science education at LUT university

Information security can be injected across the computer science curriculum in LUT university following the guidelines of the CS2013 report [30] and Vorhees et al. [86]. This can be done by taking all 37 tier 1 and tier 2 learning outcomes and injecting them across the LUT computer science courses. LUT computer science curriculum contains many courses and not all courses are a good fit for information security topics. **Table 7** shows the courses that were identified to be a natural fit for information security topics in LUT computer science curriculum.

Table 7. Selected LUT Computer Science Courses

Course name	**Abbreviation**
Basics of Database Systems	BDS
Data Structures and Algorithms	DSA
Database Systems Management	DSM
Distributed Systems	DS
Foundations of Computer Science	FCP
Foundations of Information Processing	FIP
Introduction to Programming and Data Analysis	IP & DA
Object-Oriented Programming	OOP
Ohjelmistotestauksen Periaatteet	OP
Operating Systems and System Programming	OS & SP
Principles of C-Programming	PCP
Software Engineering	SE
Web Applications	WA

Table 8 presents the 37 IAS learning outcomes and LUT computer science courses where the learning goal is a good fit. Note that many courses may be applicable for a certain learning outcome, as it's good that a student faces the same learning goals but with different course contexts. This leaves room for different levels of mastery when same

learning goal is present in multiple different courses: in some courses the students should have familiarity of the learning goal and topic, and in the next one they should learn to apply the knowledge.

Table 8 IAS Learning Outcomes for LUT courses

IAS Knowledge Unit and Learning outcomes	**Applicable Courses for Learning Outcome**
Foundational Concepts in Security	
1. Analyze the tradeoffs of balancing key security properties (Confidentiality, Integrity, and Availability)	WA, DS, OS &SP, DSM
2. Describe the concepts of risks, threats, vulnerabilities and attack vectors (including the fact that there is no such thing as perfect security)	FCP, FIP, SE, OS & SP
3. Explain the concepts of authentication, authorization, access control	WA, BDS, DSM, DS, OS & SP
4. Explain the concept of trust and trustworthiness	OS & SP, WA, DSM, BDS, FIP, PCP
5. Describe important ethical issues to consider in computer security, including ethical issues associated with fixing or not fixing vulnerabilities and disclosing or not disclosing vulnerabilities	FCP, FIP, SE, OS & SP
Principles of Secure Design	
1. Describe the principle of least privilege and isolation as applied to system design	DS, OS & SP, DSM, BDS, WA, FCP
2. Summarize the principle of fail-safe and deny-by-default	DS, WA, OS & SP, BDS
3. Discuss the implications of relying on open design or the secrecy of design for security	DSM, BDS, OS & SP, SE
4. Explain the goals of end-to-end data security	WA, DS
5. Discuss the benefits of having multiple layers of defenses	OS & SP, PCP, WA, DS, SE
6. For each stage in the lifecycle of a product, describe what security considerations should be evaluated	SE, OP
7. Describe the cost and tradeoffs associated with designing security into a product	SE, FIP, FCP
8. Describe the concept of mediation and the principle of complete mediation	SE, WA, PCP, OS & SP
9. Describe standard components for security operations, and explain the benefits of their use instead of reinventing fundamentals operations	SE
10. . Explain the concept of trusted computing including trusted computing base and attack surface and the principle of minimizing trusted computing base	DS, OS & SP
11. Discuss the importance of usability in security mechanism design	SE, WA
12. Describe security issues that arise at boundaries between multiple components.	OS & SP, DS, WA, DSM
13. Identify the different roles of prevention mechanisms and detection/deterrence mechanisms	SE, OS & SP, PCP, WA, BDS

(**Table 8 continued**)

Defensive Programming	
1. Explain why input validation and data sanitization is necessary in the face of adversarial control of the input channel	IP & DA, OS & SP, WA, BDS
2. Explain why you might choose to develop a program in a type-safe language like Java, in contrast to an unsafe programming language like C/C++	PCP, OOP, OS & SP, FCP, FIP
3. Classify common input validation errors, and write correct input validation code	IP & DA, OOP, PCP, WA, BDS, OS & SP
4. Demonstrate using a high-level programming language how to prevent a race condition from occurring and how to handle an exception	OOP, WA, BDS, OS & SP
5. Demonstrate the identification and graceful handling of error conditions	WA, OOP, OS & SP, IP & DA
6. Explain the risks with misusing interfaces with third-party code and how to correctly use third-party code	PCP, WA, OOP, OS & SP, DSM
7. Discuss the need to update software to fix security vulnerabilities and the lifecycle management of the fix	SE, WA, FCP, FIP, DS, DSM
Threats and Attacks	
1. Describe likely attacker types against a particular system	WA, SE, DS, OS & SP
2. Discuss the limitations of malware countermeasures (e.g., signature-based detection, behavioral detection)	SE, FIP, FCP, DS, DSA
3. Identify instances of social engineering attacks and Denial of Service attacks	SE, FIP, FCP, DS, WA
4. Discuss how Denial of Service attacks can be identified and mitigated	WA, DS, BDS, OS & SP
Network Security	
1. Describe the different categories of network threats and attacks	WA, DS, FIP, SE
2. Describe the architecture for public and private key cryptography and how public key infrastructure (PKI) supports network security	WA, FCP, FIP, DSA
3. Describe virtues and limitations of security technologies at each layer of the network stack	DS, OS & SP, SE
4. Identify the appropriate defense mechanism(s) and its limitations given a network threat	WA, DS, OS & SP
Crpytography	
1. Describe the purpose of cryptography and list ways it is used in data communications	DSA, WA, DS, FIP, FCP
2. Define the following terms: cipher, cryptanalysis, cryptographic algorithm, and cryptology, and describe the two basic methods (ciphers) for transforming plain text in cipher text	WA, DS, FIP, FCP, DSA
3. Discuss the importance of prime numbers in cryptography and explain their use in cryptographic Algorithms	DSA, SE, FIP, FCP
4. Explain how public key infrastructure supports digital signing and encryption and discuss the limitations/vulnerabilities	WA, DS, FIP, FCP, SE

Having the same information security learning outcome in multiple courses is intended as the topic of delivery for the same learning outcome differs course by course, and students face the same fundamentals, but in a different context throughout the curriculum. For example the defensive programming learning goals should be present in every course that contains programming assignments where programs handle input from users. In Web Applications SQL injection is topic of choice for defensive programming, and the corresponding topic is buffer overflows in Principles of C-programming.

During the courses the information security topics should be taught utilizing hands-on exercises using the offensive approach while promoting for the adversarial mindset. Teaching the students to attack makes them better prepared for defending against the attacks, as they know how the attacks work. Teaching adversarial mindset makes the students better prepared to reason when, where, how an attack will happen. For example in Web Applications course the students could be made to program an application that gets some input from the user and fetches information from the database based on the input. Then students should test each others software for injection vulnerabilities, and in presence of such vulnerabilities proposes fixes or mitigations.

Instructors can use the learning goals and learning outcomes as criteria of evaluation for practical assignments in the courses, making information security a theme that is present throughout the courses from the beginning to the end. For example in Web Applications course the instructors can look how has the student taken input validation into account by protecting against SQL injection. In the same vein in Principles of C-Programming course instructors should look how the student has protected against buffer overflows.

CTF competition can be used to test the students current knowledge of different information security topics, and to teach students more about information security. Especially if the instructors feel that the courses are already very densely packed with other topics then CTFs are a good way to present the relevant information security topics to students.

5.4 Limitations and Future Work

Firstly, the articles in this study were collected from three different scientific databases and this does not cover all published articles. Therefore it is possible that some relevant studies and useful information was missed. Secondly, white papers, books, and other non-peer-reviewed articles were excluded which might contain relevant information.

The next step would be to take the guidelines and recommendations and apply them to the LUT University's computer science education. After that the effects of the recommendations should be studied: How the recommendations were applied, has the students information security skills improved and how, and have there been any emerging ethical issues after teaching the students the offensive approach to information security.

6 CONCLUSIONS

In this thesis, what recommendations and guidelines exists for information security education, what are the learning objectives regarding information security for a computer science graduate, and how can these recommendations and guidelines be taken into account in LUT University computer science education was studied. The study was performed by conducting a systematic mapping study. Scientific databases were searched for articles related to teaching information security. A total of 65 articles were accepted and analysed. The articles were categorized based on the main issue studied. Four main categories were identified: laboratory environments, teaching specific topics, how to teach information security, and course descriptions and teaching experiences.

Information security education should spread across the computer science curriculum. The teaching should utilize virtual laboratories, so students can safely practice attacks and defences. The teaching should use the offensive approach, the students should be taught to attack so they better learn to defend. The students should be taught adversarial thinking, ie. to think like an attacker. The CS2013 report provides guidelines for computer science curriculum on information security and other topics as well.

The CS2013 report defines many learning outcomes regarding information security. Those learning outcomes can be condensed into the following learning goals:

- Ability to articulate foundational concepts in securing systems and networks,
- Ability to apply principles of secure design and defensive programming techniques.
- Ability to identify security issues with common attacks and threats.

The recommendations can be utilized in LUT University by injecting security learning goals into the existing courses. During the courses, the teaching should utilize adversarial thinking, use hands-on exercises utilizing the offensive approach.

The next step would be to take these recommendations and guidelines and apply them to the LUT University's computer science curriculum.

REFERENCES

[1] J. Fruhlinger, "What is WannaCry ransomware, how does it infect, and who was responsible?," *CSO Online*, 30-Aug-2018. [Online]. Available: https://www.csoonline.com/article/3227906/what-is-wannacry-ransomware-how-does-it-infect-and-who-was-responsible.html. [Accessed: 02-Jun-2019].

[2] A. Ozment, "Vulnerability Discovery & Software Security," University of Cambridge Computer Laboratory Computer Security Group &Magdalene College, 2007.

[3] "CWE - Common Weakness Enumeration." [Online]. Available: https://cwe.mitre.org/index.html. [Accessed: 28-Dec-2018].

[4] "NVD - CVE-2018-6789." [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2018-6789. [Accessed: 29-Dec-2018].

[5] B. L. more than 20 years *et al.*, "Malware Detection—Discovering Cross-Site Scripting Attacks," *Lastline*, 09-Nov-2017. .

[6] "The State of Open Source Vulnerability Management," *WhiteSource*. [Online]. Available: https://www.whitesourcesoftware.com/open-source-vulnerability-management-report/. [Accessed: 27-Dec-2018].

[7] H. Orman, "The Morris worm: A fifteen-year perspective," *IEEE Secur. Priv. Mag.*, vol. 1, no. 5, pp. 35–43, Sep. 2003.

[8] "NVD - Home." [Online]. Available: https://nvd.nist.gov/. [Accessed: 27-Dec-2018].

[9] "Category:OWASP Top Ten Project - OWASP." [Online]. Available: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project. [Accessed: 29-Dec-2018].

[10] "Injection Flaws - OWASP." [Online]. Available: https://www.owasp.org/index.php/Injection_Flaws. [Accessed: 29-Dec-2018].

[11] D. S. Oliveira *et al.*, "API Blindspots: Why Experienced Developers Write Vulnerable Code," p. 15.

[12] T. Unruh *et al.*, "Leveraging Flawed Tutorials for Seeding Large-Scale Web Vulnerability Discovery," p. 12.

[13] J. Wilander, "Security People vs Developers," *Apps and Security*, 13-Feb-2011. .

[14] "Why don't developers write more secure code? -," 21-Jun-2016. .

[15] "CloudPassage Study Finds U.S. Universities Failing in Cybersecurity Education," *CloudPassage*. [Online]. Available: https://www.cloudpassage.com/company/press-releases/cloudpassage-study-finds-u-s-universities-failing-cybersecurity-education/. [Accessed: 29-Dec-2018].

[16] 14:00-17:00, "ISO/IEC 27040:2015," *ISO*. [Online]. Available: http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/04/44/44404.html. [Accessed: 29-Dec-2018].

[17] "Data Breach - Definition - Trend Micro USA." [Online]. Available: https://www.trendmicro.com/vinfo/us/security/definition/data-breach. [Accessed: 29-Dec-2018].

[18] B. Edwards, S. Hofmeyr, and S. Forrest, "Hype and heavy tails: A closer look at data breaches," *J. Cybersecurity*, vol. 2, no. 1, pp. 3–14, Dec. 2016.

[19] "Data breaches 2018: Billions hit by growing number of cyberattacks," *usatoday*. [Online]. Available: http://www.usatoday.com/story/money/2018/12/28/data-

breaches-2018-billions-hit-growing-number-cyberattacks/2413411002/. [Accessed: 30-Dec-2018].
[20] "How A Cyber Breach Can Hurt Your Business," *Vantiv*. [Online]. Available: https://www.vantiv.com/vantage-point/safer-payments/data-breach-side-effects. [Accessed: 30-Dec-2018].
[21] C. Team, "Data Breaches: Threats and Consequences." [Online]. Available: https://www.cloudmask.com/blog/data-breaches-threats-and-consequences. [Accessed: 30-Dec-2018].
[22] "When Hackers Expose Cheaters: Ashley Madison Hackers Threaten to Expose User Data - Security News - Trend Micro USA." [Online]. Available: https://www.trendmicro.com/vinfo/us/security/news/cyber-attacks/when-hackers-expose-cheaters-ashley-madison-hackers-threaten-to-expose-user-data. [Accessed: 30-Dec-2018].
[23] P. Leskin, "The 21 scariest data breaches of 2018," *Business Insider*. [Online]. Available: https://www.businessinsider.com/data-hacks-breaches-biggest-of-2018-2018-12. [Accessed: 30-Dec-2018].
[24] "Data Breach Investigation Report," *Verizon Enterprise*. [Online]. Available: https://enterprise.verizon.com/resources/reports/dbir/. [Accessed: 29-Dec-2018].
[25] N. Statt, "Google hid major Google+ security flaw that exposed users' personal information," *The Verge*, 08-Oct-2018. [Online]. Available: https://www.theverge.com/2018/10/8/17951914/google-plus-data-breach-exposed-user-profile-information-privacy-not-disclosed. [Accessed: 30-Dec-2018].
[26] J. C. Wong, "Uber concealed massive hack that exposed data of 57m users and drivers," *The Guardian*, 22-Nov-2017.
[27] "2018 reform of EU data protection rules," *European Commission - European Commission*. [Online]. Available: https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en. [Accessed: 27-Dec-2018].
[28] S. D. Ringmann, H. Langweg, and M. Waldvogel, "Requirements for Legally Compliant Software Based on the GDPR," in *On the Move to Meaningful Internet Systems. OTM 2018 Conferences*, 2018, pp. 258–276.
[29] "First Hospital GDPR Violation Penalty Issued: Portuguese Hospital to Pay €400,000 GDPR Fine," *HIPAA Journal*, 07-Dec-2018. .
[30] A. for C. M. (ACM) Joint Task Force on Computing Curricula and I. C. Society, *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. New York, NY, USA: ACM, 2013.
[31] "WebOodi - Study guide - Bacherlor's Programme in Software Engineering." [Online]. Available: https://weboodi.lut.fi/oodi/vl_kehys.jsp?MD5avain=&Kieli=6&Opas=219&Org=16195577&vl_tila=1&AukAikMaar=1. [Accessed: 05-Jan-2019].
[32] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic Mapping Studies in Software Engineering," p. 10.
[33] S. Peltsverger and C. Zhang, "Bottleneck Analysis with NetKit: Teaching Information Security with Hands-on Labs," in *Proceedings of the 15th Annual Conference on Information Technology Education*, New York, NY, USA, 2014, pp. 45–50.

[34] C. Willems, T. Klingbeil, L. Radvilavicius, A. Cenys, and C. Meinel, “A distributed virtual laboratory architecture for cybersecurity training,” in *2011 International Conference for Internet Technology and Secured Transactions*, 2011, pp. 408–415.
[35] K. Salah, “Harnessing the Cloud for Teaching Cybersecurity,” in *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2014, pp. 529–534.
[36] N. Eliot, D. Kendall, and M. Brockway, “A Flexible Laboratory Environment Supporting Honeypot Deployment for Teaching Real-World Cybersecurity Skills,” *IEEE Access*, vol. 6, pp. 34884–34895, 2018.
[37] S. A. Kaabi, N. A. Kindi, S. A. Fazari, and Z. Trabelsi, “Virtualization based ethical educational platform for hands-on lab activities on DoS attacks,” in *2016 IEEE Global Engineering Education Conference (EDUCON)*, 2016, pp. 273–280.
[38] M. Timchenko and D. Starobinski, “A Simple Laboratory Environment for Real-World Offensive Security Education,” in *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2015, pp. 657–662.
[39] J. Mirkovic and T. Benzel, “Teaching Cybersecurity with DeterLab,” *IEEE Secur. Priv.*, vol. 10, no. 1, pp. 73–76, Jan. 2012.
[40] C. Tunc, S. Hariri, F. D. L. P. Montero, F. Fargo, P. Satam, and Y. Al-Nashif, “Teaching and Training Cybersecurity as a Cloud Service,” in *2015 International Conference on Cloud and Autonomic Computing*, 2015, pp. 302–308.
[41] K. Salah, M. Hammoud, and S. Zeadally, “Teaching Cybersecurity Using the Cloud,” *IEEE Trans. Learn. Technol.*, vol. 8, no. 4, pp. 383–392, Oct. 2015.
[42] R. S. Weiss, S. Boesen, J. F. Sullivan, M. E. Locasto, J. Mache, and E. Nilsen, “Teaching Cybersecurity Analysis Skills in the Cloud,” in *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2015, pp. 332–337.
[43] A. A. Jillepalli, D. C. de Leon, and F. T. Sheldon, “CERES NetSec: Hands-on Network Security Tutorials,” *J Comput Sci Coll*, vol. 33, no. 5, pp. 88–96, May 2018.
[44] R. Weiss and J. Mache, “Teaching Security Labs with Web Applications, Buffer Overflows and Firewall Configurations,” *J Comput Sci Coll*, vol. 27, no. 1, pp. 163–170, Oct. 2011.
[45] Z. Trabelsi, “Enhancing the Comprehension of Network Sniffing Attack in Information Security Education Using a Hands-on Lab Approach,” in *Proceedings of the 15th Annual Conference on Information Technology Education*, New York, NY, USA, 2014, pp. 39–44.
[46] H. Ledford, X. Mountrouidou, and X. Li, “Denial of Service Lab for Experiential Cybersecurity Learning in Primarily Undergraduate Institutions,” *J Comput Sci Coll*, vol. 32, no. 2, pp. 158–164, Dec. 2016.
[47] E. Damon, J. Dale, E. Laron, J. Mache, N. Land, and R. Weiss, “Hands-on Denial of Service Lab Exercises Using SlowLoris and RUDY,” in *Proceedings of the 2012 Information Security Curriculum Development Conference*, New York, NY, USA, 2012, pp. 21–29.
[48] Z. Trabelsi, “Hands-on Lab Exercises Implementation of DoS and MiM Attacks Using ARP Cache Poisoning,” in *Proceedings of the 2011 Information Security Curriculum Development Conference*, New York, NY, USA, 2011, pp. 74–83.

[49] Z. Trabelsi and W. Ibrahim, “Teaching ethical hacking in information security curriculum: A case study,” in *2013 IEEE Global Engineering Education Conference (EDUCON)*, 2013, pp. 130–137.
[50] Z. Trabelsi and L. Alketbi, “Using Network Packet Generators and Snort Rules for Teaching Denial of Service Attacks,” in *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education*, New York, NY, USA, 2013, pp. 285–290.
[51] Z. Trabelsi and H. Saleous, “Teaching keylogging and network eavesdropping attacks: Student threat and school liability concerns,” in *2018 IEEE Global Engineering Education Conference (EDUCON)*, 2018, pp. 437–444.
[52] T. Dimkov, W. Pieters, and P. Hartel, “Training Students to Steal: A Practical Assignment in Computer Security Education,” in *Proceedings of the 42Nd ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2011, pp. 21–26.
[53] E. Katsadouros, D. Kogias, L. Toumanidis, C. Chatzigeorgiou, and C. Z. Patrikakis, “Teaching network security through a scavenger hunt game,” in *2017 IEEE Global Engineering Education Conference (EDUCON)*, 2017, pp. 1802–1805.
[54] T. D. Nguyen and C. E. Irvine, “Development of Industrial Network Forensics Lessons,” in *Proceedings of the Fifth Cybersecurity Symposium*, New York, NY, USA, 2018, pp. 7:1–7:5.
[55] Y. Pan, D. Schwartz, and S. Mishra, “Gamified digital forensics course modules for undergraduates,” in *2015 IEEE Integrated STEM Education Conference*, 2015, pp. 100–105.
[56] Ø. Nordhaug, A. S. Imran, A. M. H. Alawawdeh, and S. J. Kowalski, “The forensic challenger,” in *2014 International Conference on Web and Open Access to Learning (ICWOAL)*, 2014, pp. 1–6.
[57] L. Simpkins, X. Yuan, J. Modi, J. Zhan, and L. Yang, “A Course Module on Web Tracking and Privacy,” in *Proceedings of the 2015 Information Security Curriculum Development Conference*, New York, NY, USA, 2015, pp. 10:1–10:7.
[58] B. Hamdan, “Teaching Case Study: Introducing Data Analytics in an Advanced Cybersecurty Course,” *J Comput Sci Coll*, vol. 33, no. 2, pp. 113–120, Dec. 2017.
[59] Z. Trabelsi and U. Mustafa, “A Web-based Firewall Simulator Tool for Information Security Education,” in *Proceedings of the Sixteenth Australasian Computing Education Conference - Volume 148*, Darlinghurst, Australia, Australia, 2014, pp. 83–90.
[60] Z. A. Wen, Y. Li, R. Wade, J. Huang, and A. Wang, “What.Hack: Learn Phishing Email Defence the Fun Way,” in *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, New York, NY, USA, 2017, pp. 234–237.
[61] J. Zhu, H. R. Lipford, and B. Chu, “Interactive Support for Secure Programming Education,” in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2013, pp. 687–692.
[62] A. Haywood, H. Yu, and X. Yuan, “Teaching Java security to enhance cybersecurity education,” in *2013 Proceedings of IEEE Southeastcon*, 2013, pp. 1–6.
[63] J. P. Landry, “Risk Assessment of Voting Systems for Teaching the Art of Information Security,” in *2010 Information Security Curriculum Development Conference*, New York, NY, USA, 2010, pp. 36–39.

[64] S. Figueroa, J. F. Carías, J. Añorga, S. Arrizabalaga, and J. Hernantes, “A RFID-based IoT Cybersecurity Lab in Telecommunications Engineering,” in *2018 XIII Technologies Applied to Electronics Teaching Conference (TAEE)*, 2018, pp. 1–8.
[65] A. Konak, T. Clark, and M. Nasereddin, “Best practices to design hands-on activities for virtual computer laboratories,” in *2013 IEEE Integrated STEM Education Conference (ISEC)*, 2013, pp. 1–7.
[66] A. Konak, T. K. Clark, and M. Nasereddin, “Using Kolb’s Experiential Learning Cycle to improve student learning in virtual computer laboratories,” *Comput. Educ.*, vol. 72, pp. 11–22, Mar. 2014.
[67] X. Yuan, I. Williams, T. H. Kim, J. Xu, H. Yu, and J. H. Kim, “Evaluating Hands-on Labs for Teaching SQL Injection: A Comparative Study,” *J Comput Sci Coll*, vol. 32, no. 4, pp. 33–39, Apr. 2017.
[68] M. Bhatt, I. Ahmed, and Z. Lin, “Using Virtual Machine Introspection for Operating Systems Security Education,” in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2018, pp. 396–401.
[69] N. Luburić, G. Sladić, J. Slivka, and B. Milosavljević, “A Framework for Teaching Security Design Analysis Using Case Studies and the Hybrid Flipped Classroom,” *ACM Trans Comput Educ*, vol. 19, no. 3, pp. 21:1–21:19, Jan. 2019.
[70] C. Chatmon, H. Chi, and W. Davis, “Active Learning Approaches to Teaching Information Assurance,” in *2010 Information Security Curriculum Development Conference*, New York, NY, USA, 2010, pp. 1–7.
[71] X. Yuan, S. Murthy, J. Xu, and H. Yu, “Case Studies for Teaching Physical Security and Security Policy,” in *2010 Information Security Curriculum Development Conference*, New York, NY, USA, 2010, pp. 21–26.
[72] J. Vaidya, D. Lorenzi, B. Shafiq, S. Chun, and N. Badar, “Teaching Privacy in an Interactive Fashion,” in *Proceedings of the 2014 Information Security Curriculum Development Conference*, New York, NY, USA, 2014, pp. 7:1–7:7.
[73] D. Ktoridou and I. Dionysiou, “Case-based learning: An instructional model to incorporate information security topcis in multidiciplinary courses at the University of Nicosia,” in *2011 IEEE Global Engineering Education Conference (EDUCON)*, 2011, pp. 466–469.
[74] Y. Cai and T. Arney, “Cybersecurity Should Be Taught Top-Down and Case-Driven,” in *Proceedings of the 18th Annual Conference on Information Technology Education*, New York, NY, USA, 2017, pp. 103–108.
[75] I. Ahmed and V. Roussev, “Peer Instruction Teaching Methodology for Cybersecurity Education,” *IEEE Secur. Priv.*, vol. 16, no. 4, pp. 88–91, Jul. 2018.
[76] W. A. Alhamdani, “Teaching Cryptography Using Design Thinking Approach,” *J. Appl. Secur. Res.*, vol. 11, no. 1, pp. 78–89, Jan. 2016.
[77] S. Peltsverger and O. Karam, “Is Teaching with Security in Mind Working?,” in *2010 Information Security Curriculum Development Conference*, New York, NY, USA, 2010, pp. 15–20.
[78] J. McManus, “Security by Design: Teaching Secure Software Design and Development Techniques,” *J Comput Sci Coll*, vol. 33, no. 3, pp. 75–82, Jan. 2018.
[79] S. Bratus, A. Shubina, and M. E. Locasto, “Teaching the Principles of the Hacker Curriculum to Undergraduates,” in *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2010, pp. 122–126.

[80] M. Mink and R. Greifeneder, “Evaluation of the Offensive Approach in Information Security Education,” in *Security and Privacy – Silver Linings in the Cloud*, 2010, pp. 203–214.
[81] B. Wilson, “Teaching Security Defense Through Web-based Hacking at the Undergraduate Level,” *J Comput Sci Coll*, vol. 33, no. 2, pp. 121–128, Dec. 2017.
[82] G. Conti and J. Caroland, “Embracing the Kobayashi Maru: Why You Should Teach Your Students to Cheat,” *IEEE Secur. Priv.*, vol. 9, no. 4, pp. 48–51, Jul. 2011.
[83] S. T. Hamman, K. M. Hopkinson, R. L. Markham, A. M. Chaplik, and G. E. Metzler, “Teaching Game Theory to Improve Adversarial Thinking in Cybersecurity Students,” *IEEE Trans. Educ.*, vol. 60, no. 3, pp. 205–211, Aug. 2017.
[84] Y. Vorobeychik *et al.*, “Fireaxe: The DHS Secure Design Competition Pilot,” in *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, New York, NY, USA, 2013, pp. 21:1–21:4.
[85] A. Joshi, V. Ramani, H. Murali, R. Krishnan, Z. Mithra, and V. Pavithran, “Student centric design for cyber security knowledge empowerment,” in *2012 IEEE International Conference on Technology Enhanced Education (ICTEE)*, 2012, pp. 1–4.
[86] D. Voorhees, A. Das, and C. Choi, “Injecting and Assessing Cybersecurity Topics Within a Computer Science Program,” *J Comput Sci Coll*, vol. 32, no. 6, pp. 54–66, Jun. 2017.
[87] J. T. McDonald and T. R. Andel, “Integrating Historical Security Jewels in Information Assurance Education,” *IEEE Secur. Priv.*, vol. 10, no. 6, pp. 45–50, Nov. 2012.
[88] L. González-Manzano and J. M. de Fuentes, “Design recommendations for online cybersecurity courses,” *Comput. Secur.*, vol. 80, pp. 238–256, Jan. 2019.
[89] A. V. Uskov, “Applied Cryptography for Computer Science programs: A practitioner’s approach,” in *2013 3rd Interdisciplinary Engineering Design Education Conference*, 2013, pp. 63–70.
[90] N. Meghanathan, H. Kim, and L. A. Moore, “Incorporation of Aspects of Systems Security and Software Security in Senior Capstone Projects,” in *2012 Ninth International Conference on Information Technology - New Generations*, 2012, pp. 319–324.
[91] A. V. Uskov, “Hands-On Teaching of Software and Web Applications Security,” in *2013 3rd Interdisciplinary Engineering Design Education Conference*, 2013, pp. 71–78.
[92] Y. Wang, M. McCoey, and H. Zou, “Developing an Undergraduate Course Curriculum on Information Security,” in *Proceedings of the 19th Annual SIG Conference on Information Technology Education*, New York, NY, USA, 2018, pp. 66–71.
[93] V. Švábenský, J. Vykopal, M. Cermak, and M. Laštovička, “Enhancing Cybersecurity Skills by Creating Serious Games,” in *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, New York, NY, USA, 2018, pp. 194–199.
[94] J. R. Aman, J. E. Conway, and C. Harr, “A Capstone Exercise for a Cybersecurity Course,” *J Comput Sci Coll*, vol. 25, no. 5, pp. 207–212, May 2010.
[95] R. K. Raj and R. Savacool, “Experiences with teaching secure data management,” in *2010 IEEE Frontiers in Education Conference (FIE)*, 2010, pp. S3F-1-S3F-6.

[96] V. Jovanovic and T. Mirzoev, “Teaching Storage Infrastructure Management and Security,” in *2010 Information Security Curriculum Development Conference*, New York, NY, USA, 2010, pp. 41–44.
[97] W. Wei and L. Sun, “Innovative Teaching and Learning Experiences in Network Attacks and Defense,” in *2012 Eighth International Conference on Computational Intelligence and Security*, 2012, pp. 587–590.
[98] W. Sun, “Experiences from a Time-Condensed Computer Security Class,” in *2010 Seventh International Conference on Information Technology: New Generations*, 2010, pp. 482–487.

APPENDIX 1. Table of accepted articles

Main issue studied	Publication year	Reference
Presents virtualization laboratory environment	2014	[33]
Distributed virtualization laboratory architecture for teaching information security	2011	[34]
Using cloud laboratories in teaching information security	2014	[35]
Presents a physical laboratory that supports honepots	2018	[36]
Virtualization laboratory for teaching DOS attacks	2016	[37]
Simple virtualization laboratory environment for teaching offensive security	2015	[38]
Presents DeterLab virtualization platform for information security assignments	2012	[39]
Presents ClaaS cloud-based virtula laboratory environment	2015	[40]
Using cloud laboratories in teaching information security	2015	[41]
Present EDURange framework, cloud service for teaching information security	2015	[42]

APPENDIX 1. (continues)

Tutorials for teaching network security	2018	[43]
Presents hands-on exercises for teaching information security	2011	[44]
Teaching network eavesdropping attacks	2014	[45]
Teaching DOS attacks	2016	[46]
Teaching DOS attacks	2012	[47]
Teaching DOS attacks	2011	[48]
Teaching DOS attacks	2013	[49]
Teaching DOS attacks	2013	[50]
Teaching network eavesdropping attacks	2018	[51]
Presents a physical penetration testing assignment	2011	[52]
Teaching wireless network security using a game	2017	[53]
Teaching digital forensics	2018	[54]
Teaching digital forensics	2015	[55]
Teaching digital forensics	2014	[56]
Course module on web privacy issues	2015	[57]
Teaching data analytics for intrusion detection	2017	[58]
Describes Edu-Firewall for teaching firewall rules and concepts	2014	[59]
Presents What.Hack game for teaching identification of phising attacks	2017	[60]

APPENDIX 1. (continues)

Teaching secure programming with IDE plugin	2013	[61]
Teaching secure programming with Java	2013	[62]
Hands-on case study assignment for teaching for teaching risk assessment	2010	[63]
Teaching RFID security analysis	2018	[64]
Framework for designing hands-on laboratory assignments	2013	[65]
Framework for designing hands-on laboratory assignments	2014	[66]
Evaluates hands-on laboratory assignment presentations methods	2017	[67]
Framework for teaching kernel security	2018	[68]
Framework for teaching security design analysis	2019	[69]
Active learning approaches for teaching information security	2010	[70]
Case studies for teaching physical security	2010	[71]
Using case studies and game-based learning for teaching privacy	2014	[72]

APPENDIX 1. (continues)

Using case studies to integrate information security topics in computer scienc courses	2011	[73]
How information security should be taught case-driven and top-down	2017	[74]
A peer-instructtion methodology for information security education	2018	[75]
Teaching model for teaching cryptography based on design thinking approach	2016	[76]
Evaluation of teaching security first approach	2010	[77]
Teaching secure software design and development	2018	[78]
Teaching Hacker Curriculum to students	2010	[79]
Evaluates offensive vs defensive teaching approach	2010	[80]
Teaching security defense by teaching students to attack	2017	[81]
Experiences teaching adversarial thinking	2011	[82]
Teaching adversarial thinking with game theory	2017	[83]
Using CTF to teach information security	2013	[84]
Using CTF to teach information security	2012	[85]

APPENDIX 1. (continues)

Injecting information security accross curriculum	2017	[86]
Teaching specific information security principles	2012	[87]
Recommendations for designing information security MOOCs	2019	[88]
Course description for applied cryptography	2013	[89]
Experiences incorporating information security aspects into capstone projects	2012	[90]
Presents a course description on software and web application security	2013	[91]
Presents an information security course curriculum	2018	[92]
Describes information security courses utilizing games in teaching	2018	[93]
Describes a capstone exercise for information security course	2010	[94]
Experiences teaching secure data management	2010	[95]
Experiences teaching secure data management	2010	[96]
Experiences teaching a network security course	2012	[97]
Experiences teaching a time-condensed course	2010	[98]