The Complete Study Guide for DT145G Computer Security

Daniel Bosk

Department of Information Systems and Technology
Mid Sweden University, Sundsvall

School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology, Stockholm

28th February 2020

Contents

1 Scope and aims

This course is an introduction to computer security. The course aims towards a good understanding for the requirements of a secure computer system. Problems such as authentication and access control; software security, such as buffer overflows; as well as operating system, library and application security mechanisms are treated in the course.

More concretely, the intended learning outcomes (ILOs) of the course are the following.

After completing the course, you should be able to:

- *analyse* a security problem and *propose* a solution to it.
- *value and argue* about different ethical aspects of computer security.

This requires you to be able to:

- *categorize* a research question into what research method is suitable to answer it.
- *apply* different cryptographic primitives and *explain* how these work (on a high level),
- *analyse* problems of authentication, access control, accountability and different solutions,
- *explain* how some common attacks on software works and *analyse* code for security vulnerabilities,
- *evaluate* strengths and weaknesses of hardware-based security such as full-disk encryption, as well as

The course has a variety of learning sessions designed to ensure that you learn these ILOs. Each such session has a set of further specified ILOs that will help you achieve the ILOs above.

The grades will be based on the following grading criteria.

Grade E You fulfil all the ILOs above. You should have identified a relevant problem, and given a solution to it. It must be a viable solution, however gaps and mistakes are allowed, if they don't render your solution unusable.

Grade C You fulfil the criteria for E. Additionally, your evaluations and designs are *good* with *some base* in theory and, where applicable, the research literature. Gaps and errors are allowed if they only render your solution less optimal.

Grade A You fulfil the criteria for C. However, your evaluations and designs must be *extensive* and *well-founded* in theory and, where applicable, the research literature. Gaps and errors are not allowed in the solution unless they have been properly addressed and you have given a suggestion on an approach to how to start resolve the issue.

The grades B and D are intermediary grades.

2 Course structure and content overview

The course covers applied cryptography used in computer security, e.g., uses of cryptography for code obfuscation or digital rights management; authentication mechanisms, access control, and intrusion detection; software security, e.g., buffer overruns and interaction between programs; some security mechanisms provided by operating system and hardware; and malicious software and how these utilise weaknesses in the system. Finally, we discuss some ethical implications for computer engineers.

2.1 Teaching and tutoring

The course takes a flipped-classroom approach. This means that there will be pre-recorded lectures and the classroom time will be spent where it is needed the most. The videos are available through ScalableLearning. There you can post questions related to the content during the video, you might answer quizzes in the video etc. These questions and the results to any quizzes will be available to the teacher and the teacher will review these before the classroom session. After reviewing any difficulties, the classroom time will be spent working with the material.

Some sessions are mandatory. The sessions for seminars are mandatory, you will see this in the assignment instructions. All assignments are numbered consecutively prefixed with an 'L' for laboratory assignments, 'S' for seminar assignments.

2.2 Schedule

You will find an outline for a schedule for the course in Table 1. You are free to follow this schedule or any schedule you make for yourself, but the learning and tutoring sessions, deadlines etc. will follow this schedule. The detailed reading instructions for each item in the schedule can be found in the following sections.

3 Course content

This section summarizes the material covered by the lectures and assignments, i.e., what you should read for each of them. It is divided by topics and ordered according to progression of the course, Table 1 gives an overview along with a schedule.

3.1 S0 What's up with security?

Summary: The purpose of this assignment is to get an idea of how security affects products, which in turn affects not only the companies behind them, but also the consumers and can have effects on a societal scale.

Intended learning outcomes: The aim of this assignment is

- to *reflect* on the effects of security, or lack thereof, on both individual and society.
- to *value and argue* about the responsibilities of engineers.

Week	**Work**
1	Lecture: Course start/Introduction Seminar S0: What's up with security? (§ 3.1)
2	Session: Foundations and usability (§ 3.2)
3	Session: Info theory (§ 3.3) Session: 'Normal' Crypto Session: Zero-knowledge and multiparty computation
4	Session: Authentication (§ 3.4) Seminar L1: pwdeval, session 1 Seminar L1: pwdeval, session 2
5	Seminar L1: pwdeval, session 3 Session: Protocols (§ 3.5) Seminar L2: pricomlab
6	Session: Access control (§ 3.6) Session: Trusted computing (§ 3.7)
7	Session: Accountability (§ 3.8) Session: Distributed Ledger Technologies
8	Session: Software security (§ 3.9)
9	Seminar L3: tools (§ 3.10) Lecture: Course conclusion (§ 3.11)
10	Exam: course exam Seminars: second call for seminars
+3 months	Exam: re-exam Seminar: final call for seminars
+6 months	Exam: last re-exam until next year

Table 1: A summary of the parts of the course and when they will (or should) be done. The table is adapted to taking this course at half-time pace, i.e., 20 hours per week for 10 weeks.

Reading: To be able to reason and have a discussion, we will have some ethics guidelines as a base: *Code of Ethics: ACM Code of Ethics and Professional Conduct* [1], *Software Engineering Code of Ethics and Professional Practice* [2] and *IEEE Code of Ethics* [3].

First, you must read up on the influence campaigns during the 2016 US election [4]. Then you must read up on the Cambridge Analytica scandal [e.g., 5–8] and the Mirai botnet incident [9].

Finally, you should search for and read current news articles of your own choice illustrating the problem of lacking security.

3.2 Foundations

What is security? *Summary:* In this learning session we will cover the foundations of security. By this we mean what security is all about, e.g., what types of properties we are interested in and what we want to achieve in our security work.

Intended learning outcomes: After this session you should be able:

- to *understand* the what security is generally about.

Reading: You should read

- Chapter 3, 'Foundations of Computer Security' of [10]. There Gollmann attempts at a definition of Computer Security and related terms, e.g., confidentiality, integrity, and availability, which we need for our treatment of the topic.
- Chapter 1 of [11]. Anderson treats a wider area than just *computer* security, which is good for us, he covers many aspects of security in different examples.

The scientific method *Summary:* In this learning session we will give an introduction to the scientific method and particularly how this can be applied in the area of security.

Intended learning outcomes: After this session you should be able:

- to *differentiate* which types of scientific methods are appropriate to answer a given question.

Reading: You should read

- 'How to Design Computer Security Experiments' [12], this paper discusses the scientific method of (parts of) the security field.
- 'SoK: Science, Security and the Elusive Goal of Security as a Scientific Pursuit' [13], for a more both wider and more in-depth reflection on the state of security as a scientific pursuit.

Attacking humans *Summary:* One important aspect of security is users' weaknesses. There are many ways to attack systems through their human operators. During this learning session we cover a variety of examples of such attacks.

Intended learning outcomes: After this learning session you should be able:

- to *adopt* an adversarial thinking for situtions involving humans.

Reading: Anderson gives a short summary of the psychology of users, their strengths and weaknesses, in Chapter 2 "Usability and Psychology" of *Security Engineering* [11].

Psychology *Summary:* One important aspect of security, which technical people tend to forget, is the users' weaknesses. The psychology of the human mind is therefore an important subject to discuss in the context of security. And consequently, we must adapt our systems to those limitations. In this learning session, we will focus on relevant parts of our psychology.

Intended learning outcomes: After this learning session you should be able:

- to *incorporate* basic psychology in the design of a system to increase its security.

Reading: Anderson gives a short summary of the psychology of users, their strengths and weaknesses, in Chapter 2 'Usability and Psychology' of *Security Engineering* [11].

3.3 Cryptography

Basic information theory The area of Information Theory was founded in 1948 by Claude Shannon. It is a mathematical theory to reason about how much information is contained in certain data. Equivalently, it is also a measure of uncertainty in information, and has thus plenty of application in security and cryptography. This learning session covers the basic concept, Shannon entropy, and some applications to security and privacy.

After the session you should be able

- to *apply* Shannon entropy in basic situations related to security and privacy.

The concept of Shannon entropy, the main part of information theory, is treated in a few short texts: *A Primer on Information Theory and Privacy* [14] and 'Chapter 6: Shannon entropy' [15]. You should read on the use of entropy to estimate identifiability: 'How Unique Is Your Browser?' [16].

A high-level overview of crypto Cryptography has a central role in security. To fully understand how many security mechanisms can be implemented we need cryptography. For this reason, we also need higher-level knowledge about what can be achieved with cryptography to not limit our thoughts about possible solutions. This learning session is intended to give a high-level overview of cryptography: symmetric-key encryption (SKE), public-key encryption (PKE), digital signatures, zero-knowledge proof (ZKP) and secure multiparty computation (MPC). In particular, the ILOs are that you should be able to

- *understand* what properties can be achieved with cryptography.
- *analyse* a situation and *suggest* what cryptographic properties are desirable.

The basics are covered by Chapter 5 in Anderson's *Security Engineering* [11] and Chapter 14 in Gollmann's *Computer Security* [10]. (To practice your understanding of these mechanisms it is recommended to do exercises 14.2, 14.3 and 14.7 in [10].) For the remaining topics, however, we refer to the *Encyclopedia of cryptography and security* [17] (and cited papers and books).

3.4 Authentication

Authentication is part of the core of security. An entity claims something, a property or an identity, authentication is about verifying or rejecting any such claim. We will discuss three aspects of authentication: user-to-machine (and user-to-user), machine-to-user, machine-to-machine. For user authentication we will start with the traditional something you know, something you have and something you are and then look beyond.

More specifically, the session should prepare you to be able to

- *understand* the authentication and usability problems of authentication involving users.
- *analyse* the requirements for authentication in a situation and *design* an authentication system with desired authentication properties and usability.

Why we want to do this and how we can accomplish this is treated in Chapter 4 in [10]. Anderson also treats this topic [11, Chap. 2], although in a wider perspective with less technical details. When you have studied this material you should do exercises 4.2, 4.3, 4.4 and 4.6 in [10]. For the treatment of anonymous credentials, we refer to 'Electronic Identities Need Private Credentials' [18] and 'Anon-Pass: Practical Anonymous Subscriptions' [19].

3.4.1 L1 Evaluating and designing authentication

A lot of user authentication is based on passwords. We use password policies to aid users in selecting a secure password. Unfortunately, research has shown that the common password-polices do not have the expected effect: users can still choose easy-to-guess passwords and the policies actually makes guessing easier. It is thus important to *scientifically* evaluate the actual effects of any user-authentication mechanism, otherwise our security might be at risk. Here we will focus on exactly that. More specifically, after this lab you should be able to

- *evaluate* the effective security by considering security and usability.
- *analyse* research results in usable security and *apply* those relevant to a given situation.
- *design* security policies aligned with usability.

To do this, we must be familiar with several topics: usability [11, Ch. 2], cryptography [11, Ch. 5] [20], information theory [15] and the scientific method [21]. The main contents is some research papers on password security and usability: 'Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms' [22], 'Of passwords and people: Measuring

the effect of password-composition policies' [23], 'Can long passwords be secure and usable?' [24] and 'The Password Life Cycle' [25]; complemented by a paper on the usability of password managers: 'A comparative usability evaluation of traditional password managers' [26].

3.5 Protocols

As soon as two entities need to interact, there is need for a protocol — be it inside or between systems, even one entity communicating with itself in different points in time (which is the case when storing something for use at a later time). These protocols need different properties. We will explore how to design secure protocols and introduce some tools for verifying security properties of protocols.

More concretely, after this session you should be able to

- *overview* the different approaches and their limits to verify the security of protocols.

Anderson gives an overview of this area in *Security Engineering* [11], Chapter 3 'Protocols'. Gollmann has a more technically oriented treatment of a part of this topic in Chapter 15 of *Computer Security* [10].

3.5.1 L2 Private communication

The more our society depends on digital systems, the more important private communication becomes. We need private communications to sustain democracy, thus we need it to be available to everyone. The purpose of this laboratory work is to introduce some practical aspects of private messaging. More specifically, after it, you should be able to

- *apply* (securely!) some common implementations of cryptography for private communication — also including any set-up (e.g. key verification).
- *analyse* different systems for private communication based on their security properties and *evaluate* which is suitable in a given situation.
- *evaluate* different implementations of private communication from a usability perspective.

The topics of this assignment are: usability [11, Ch. 2] and cryptography [11, Ch. 5] and privacy-enhancing technologies (PETs) [11, Ch. 23.4]. We then rely on the 'Why Johnny can't encrypt' papers:

- 'Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0.' [27],
- 'Why Johnny still can't encrypt: Evaluating the usability of email encryption software' [28],
- 'Why Johnny still, still can't encrypt: Evaluating the usability of a modern PGP client' [29],
- 'Can Johnny finally encrypt?: evaluating E2E-encryption in popular IM applications' [30].

3.6 Access control

Once you have authenticated users you can support access control — and this is also one of the main reasons to authenticate them in the first place. Access control aims at controlling who may access what and how they may access it. There are different models and ways to implement access control. Here we will give an overview of the possibilities. In particular, the ILOs are that you are able to:

- *understand* the fundamental access control models and their relations.
- *evaluate* advantages and disadvantages of different access control solutions.
- *analyse* a situation and *design* a proper access control solution.

The reading material is Chapter 5, followed by Chapters 11 and 12, in *Computer Security* [10]. Anderson also treats the subject in Chapters 4, 8, and 9 of *Security Engineering* [11]. (Only one of the two books is necessary to read.)

3.7 Trusted computing

Summary: One can only do so much with software. One problem with software and general purpose processors is that the software can be modified and the processor will still execute it. Another is that, that running software cannot evaluate the processing environment which executes it.

Some examples: Alice had her laptop in her bag as it passed through the security check. While she was busy with the scans, one customs official booted the laptop from a USB stick and installed a different boot loader. Or, how can Alice even trust the computer when it is brand new? Another aspect of this is to protect parts of the system from Alice herself, e.g., this is what digital rights management (DRM) is all about. We also have the compartmentalization of apps in a smartphone. If Alice accidentally installs a malicious app, it shouldn't be able to compromize the banking app. Here we will explore how to ensure the integrity of the computer system.

Intended learning outcomes: More concretely, after this session you should be able to

- *understand* the problem of trusted computing, its approaches to solutions, the underlying assumptions and its limitations.
- *analyse* different approaches to trusted computing and their limitations and *apply* them in a solution to a given problem.

Reading: We touch on the topics in Chapters

- 4 (4.2.11–4.4.1),
- 16,
- 17,
- 18 (read until and including 18.2.1) and
- 23 (23.1–23.2)

in *Security Engineering* [11].

For root-of-trust, there is the paper [31] by Gligor and Woo. Sections I and II are enough (we don't need more than an overview). Malenkovich [32] and Mimoso [33, 34] provides some examples of real-world problems in this area.

For trusted execution-environments, we use Intel SGX as an example. This is introduced by M. and O. [35]. (For a very detailed exposition on SGX, see the work by Costan and Devadas [36].)

3.8 Accountability

The need for accountability has been apparent in civilisations for as long as they have existed. One of today's institutions which is historically renowned for keeping strict accounts is the state tax office, another is, of course, the banks. We will explore some principles in keeping accounts and discuss ways to implement it in different, sometimes challenging, environments. In particular, the ILOs are that you are able to:

- *evaluate* advantages and disadvantages of different levels of accountability.
- *analyse* a situation and *design* proper accountability and, in particular, with privacy considerations.

Anderson describes accountability through his experience from banks in Chapter 10 'Banking and Bookkeeping' in *Security Engineering* [11]. We will also use the secure logging system of Schneier and Kelsey [37] as an example of how to achieve secure logging in a challenging environment. The construction described therein is a method to safely store audit logs in an untrusted machine; in the scheme, all log entries generated prior to a compromise will be impossible for the attacker to read, modify, or destroy undetectably.

3.9 Software security

Perhaps the part of security most people intuitively associate with security, and computer security in particular, is software security. This part of computer security treats vulnerabilities in software, e.g. buffer overruns or code injections. This is a very important part of security, because although the design is flawless, its implementation might have vulnerabilities. As an example, most phones are designed to keep the user and applications unpriviledged, thus all applications will run with the principle of least priviledges and compartmentalized from each other. However, software bugs in the operating system can allow malicious apps to gain priviledges to e.g. monitor other apps.

After this session you should be able to

- *understand* the need to consider software security in software development.
- *evaluate* the software security requirements for different sitations.

Gollmann treats this area in Chapter 10 of his book, *Computer Security* [10]. The recommended exercises to do after reading this material are 10.1, 10.3 and 10.4 in [10]. Anderson also treats this subject — in Chapter 4.4 and Chapter 18 of *Security Engineering* [11] — albeit with less technical details. We also treat the results of 'Four Software Security Findings' [38].

LADOK	Credits (ECTS)	Grade	Course Assignments
I104	0.0	P, F	S0
L104	3.0	P, F	L1, L2, L3
S104	1.5	P, F	S0
T104	3.0	A–F	Exam
Total	7.5	A–F	(Determined by exam)

Table 2: Table summarizing course modules and their mapping to LADOK. P means pass, F means fail. A–E are also passing grades, where A is the best.

3.10 L3 Tools of the trade

Before starting this assignment you must have a wide grasp of the theory of security. If you do not, then you will not know of all available mechanisms. Hence you will neither know of all practicalities you will have to solve to use these as a developer.

3.11 Course conclusion

During this lecture we will shortly review the course and try to fit things into a bigger picture. This is also a chance for revision and final questions before the exam.

3.12 Final exam

The final exam will assess how well you have achieved the intended learning outcomes of the course. Hence, it covers all the content given above.

Each question on the exam covers one topic of the course. To pass the exam (and thus the course) you must pass all questions (thus all topics), i.e., *you must not receive zero on any question.* If you receive a *zero on one question* you qualify for complementing that zero orally during a meeting.

4 Assessment

This section explains how the course modules are graded and mapped to LADOK. Table 2 visualizes the relations between modules, credits, grades and LADOK.

The written exam will be graded A–E for passing grades, F or Fx for failing grades. You will receive an Fx if you are very close to passing. In this case you may complement your written exam with an oral exam. If you do not take this chance you must retake the exam the next time it is given. The grade of the exam will also be the grade of the course total.

4.1 Handed-in assignments

In general, all hand-ins in the course must be in a 'passable' condition; i.e., they must be well-written, grammatically correct and without spelling errors, have citations and references according to [**IEEEcitation**] (see also [**PurdueCitation**]

for a tutorial), and finally fulfil all requirements from the assignment instruction. If you hand something in which is not in this condition, you will receive an F without further comment.

All material handed-in must be created by yourself, or, in the case of group assignments, created by you or one of the group members. When you refer to or quote other texts, then you must provide a correct list of references and, in the case of quotations, the quoted text must be clearly marked as quoted. If any part of the document is plagiarized you risk being suspended from study for a predetermined time, not exceeding six months, due to disciplinary offence. If it is a group assignment, all group members will be held accountable for disciplinary offence unless it is clearly marked in the work who is responsible for the part containing the plagiarism.

If cooperation takes place without the assignment instruction explicitly allowing this, this will be regarded as a disciplinary offence with the risk of being suspended for a predetermined time, not exceeding six months. Unless otherwise stated, all assignments are to be done individually.

4.2 'What if I'm not done in time?'

The deadlines on this course are of great importance, make sure to keep these! You must have completed the introductory assignment within its deadline. If you do not do this you will be deregistered from the course and your place will be open to other students.

For seminars and presentations there will be three sessions during the course of a year, if you cannot make it to any of those you will have to return the next time the course is given; i.e., up to a year later. All of these sessions will be in the course schedule (in the Student Portal). If you miss a deadline for the preparation for a seminar session, then you have to go for the next seminar even if the first seminar has not passed yet.

Written assignments are graded once during the course, most often shortly after the deadline of the assignment. After the course you are offered two more attempts within a year. In total you have three chances for having your assignments graded over the period of a year. After that you should come back the next time the course is given.

No tutoring is planned after the end of the course, i.e., after the last tutoring session scheduled in the course schedule. If you are not done with your assignments during the course and want to be guaranteed tutoring you have to reregister for the next time the course is given. Reregistration is a lower priority class of applicants for a course, all students applying for the course the first time have higher priority — this includes reserves too.

Thus, if you feel that you will not be done with the course on time, it is better to stop the course at an early stage. If you register a break within three weeks of the course start, you will be in the higher priority class of applicants the next time you apply for the course. You can register such a break yourself in the Student Portal.

References

[1] Association for Computing Machinery. *Code of Ethics: ACM Code of Ethics and Professional Conduct*. Accessed on 4 April 2014. URL: https://www.acm.org/about/code-of-ethics.

[2] Association for Computing Machinery. *Software Engineering Code of Ethics and Professional Practice*. Accessed on 27 March 2019. URL: https://ethics.acm.org/code-of-ethics/software-engineering-code/.

[3] Institute of Electrical and Electronics Engineers. *IEEE Code of Ethics*. Accessed on 4 April 2014. URL: http://www.ieee.org/about/corporate/governance/p7-8.html.

[4] Scott Shane and Mark Mazzetti. 'Inside a 3-Year Russian Campaign to Influence U.S. Voters'. en-US. In: *The New York Times* (Nov. 2018). ISSN: 0362-4331. URL: https://www.nytimes.com/2018/02/16/us/politics/russia-mueller-election.html (visited on 21/01/2019).

[5] Andrea Valdez. 'Everything You Need to Know About Facebook and Cambridge An alytica'. In: *Wired* (Mar. 2018). ISSN: 1059-1028. URL: https://www.wired.com/story/wired-facebook-cambridge-analytica-coverage/ (visited on 17/01/2019).

[6] Carole Cadwalladr and Emma Graham-Harrison. 'Revealed: 50 million Facebook profiles harvested for Cambridge Anal ytica in major data breach'. en-GB. In: *The Guardian* (Mar. 2018). ISSN: 0261-3077. URL: https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-%20influence-us-election (visited on 17/01/2019).

[7] Matthew Rosenberg, Nicholas Confessore and Carole Cadwalladr. 'How Trump Consultants Exploited the Facebook Data of Millions '. en-US. In: *The New York Times* (Apr. 2018). ISSN: 0362-4331. URL: https://www.nytimes.com/2018/03/17/us/politics/cambridge-analytica-trump-c%20ampaign.html (visited on 17/01/2019).

[8] Ishaan Tharoor. *Analysis — The scary truth that Cambridge Analytica under stands*. en. 2018. URL: https://www.washingtonpost.com/news/worldviews/wp/2018/03/22/the-scary-tru%20th-that-cambridge-analytica-understands/ (visited on 17/01/2019).

[9] Bruce Schneier. *Lessons From the Dyn DDoS Attack - Schneier on Security*. 2016. URL: https://www.schneier.com/blog/archives/2016/11/lessons_from_th_5.html (visited on 17/01/2019).

[10] Dieter Gollmann. *Computer Security*. 3rd ed. Chichester, West Sussex, U.K.: Wiley, 2011. ISBN: 9780470741153 (pbk.)

[11] Ross J. Anderson. *Security Engineering. A guide to building dependable distributed systems*. 2nd ed. Indianapolis, IN: Wiley, 2008. ISBN: 978-0-470-06852-6 (hbk.) URL: http://www.cl.cam.ac.uk/~rja14/book.html.

[12] Sean Peisert and Matt Bishop. 'How to Design Computer Security Experiments'. In: *Fifth World Conference on Information Security Education*. Ed. by Lynn Futcher and Ronald Dodge. Boston, MA: Springer US, 2007, pp. 141–148. ISBN: 978-0-387-73269-5.

[13] C. Herley and P. C. v. Oorschot. 'SoK: Science, Security and the Elusive Goal of Security as a Scientific Pursuit'. In: *2017 IEEE Symposium on Security and Privacy (SP)*. May 2017, pp. 99–120. DOI: 10.1109/SP.2017.38.

[14] Peter Eckersley. *A Primer on Information Theory and Privacy*. Jan. 2010. URL: https://www.eff.org/deeplinks/2010/01/primer-information-theory-and-privacy.

[15] Daniel Ueltschi. 'Chapter 6: Shannon entropy'. URL: http://www.ueltschi.org/teaching/chapShannon.pdf.

[16] Peter Eckersley. 'How Unique Is Your Browser?' In: *Privacy Enhancing Technologies*. Springer. 2010, pp. 1–18. URL: https://panopticlick.eff.org/static/browser-uniqueness.pdf.

[17] Henk CA Van Tilborg and Sushil Jajodia. *Encyclopedia of cryptography and security*. Springer Science & Business Media, 2011. URL: https://link.springer.com/referencework/10.1007%2F978-1-4419-5906-5.

[18] J. Camenisch, A. Lehmann and G. Neven. 'Electronic Identities Need Private Credentials'. In: *IEEE Security Privacy* 10.1 (Jan. 2012), pp. 80–83. ISSN: 1540-7993. DOI: 10.1109/MSP.2012.7.

[19] M. Z. Lee, A. M. Dunn, J. Katz, B. Waters and E. Witchel. 'Anon-Pass: Practical Anonymous Subscriptions'. In: *IEEE Security Privacy* 12.3 (May 2014), pp. 20–27. ISSN: 1540-7993. DOI: 10.1109/MSP.2013.158.

[20] Daniel Bosk. 'A high-level overview of cryptography'. Lecture. 2016. URL: https://github.com/OpenSecEd/appliedcrypto/releases/tag/v1.1.

[21] Sean Peisert and Matt Bishop. 'How to Design Computer Security Experiments'. In: *Fifth World Conference on Information Security Education: Proceedings of the IFIP TC11 WG 11.8, WISE 5, 19 to 21 June 2007, United States Military Academy, West Point, New York, USA*. Ed. by Lynn Futcher and Ronald Dodge. Boston, MA: Springer US, 2007, pp. 141–148. ISBN: 978-0-387-73269-5. DOI: 10.1007/978-0-387-73269-5_19. URL: http://web.cs.ucdavis.edu/~peisert/research/Peisert-WISE2007-SecurityExperiments.pdf.

[22] Patrick Gage Kelley, Saranga Komanduri, Michelle L Mazurek, Richard Shay, Timothy Vidas, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor and Julio Lopez. 'Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms'. In: *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE. 2012, pp. 523–537. DOI: 10.1109/SP.2012.38.

[23] Saranga Komanduri, Richard Shay, Patrick Gage Kelley, Michelle L. Mazurek, Lujo Bauer, Christin Nicolas, Lorrie Faith Cranor and Serge Egelman. 'Of passwords and people: Measuring the effect of password-composition policies'. In: *CHI*. 2011. URL: http://cups.cs.cmu.edu/rshay/pubs/passwords_and_people2011.pdf.

[24] Richard Shay, Saranga Komanduri, Adam L Durity, Phillip Seyoung Huh, Michelle L Mazurek, Sean M Segreti, Blase Ur, Lujo Bauer, Nicolas Christin and Lorrie Faith Cranor. 'Can long passwords be secure and usable?' In: *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. ACM. 2014, pp. 2927–2936. URL: http://lorrie.cranor.org/pubs/longpass-chi2014.pdf.

[25] Elizabeth Stobert and Robert Biddle. 'The Password Life Cycle'. In: *ACM Trans. Priv. Secur.* 21.3 (Apr. 2018), 13:1–13:32. ISSN: 2471-2566. DOI: 10.1145/3183341.

[26] Ambarish Karole, Nitesh Saxena and Nicolas Christin. 'A comparative usability evaluation of traditional password managers'. In: *International Conference on Information Security and Cryptology*. Springer. 2010, pp. 233–251.

[27] Alma Whitten and J Doug Tygar. 'Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0.' In: *USENIX Security Symposium*. Vol. 348. 1999.

[28] Steve Sheng, Levi Broderick, Colleen Alison Koranda and Jeremy J Hyland. 'Why Johnny still can't encrypt: Evaluating the usability of email encryption software'. In: *Symposium On Usable Privacy and Security*. 2006, pp. 3–4.

[29] Scott Ruoti, Jeff Andersen, Daniel Zappala and Kent Seamons. 'Why Johnny still, still can't encrypt: Evaluating the usability of a modern PGP client'. In: *arXiv preprint arXiv:1510.08555* (2015).

[30] Amir Herzberg and Hemi Leibowitz. 'Can Johnny finally encrypt?: evaluating E2E-encryption in popular IM applications'. In: *Proceedings of the 6th Workshop on Socio-Technical Aspects in Security and Trust*. ACM. 2016, pp. 17–28.

[31] Virgil D Gligor and Shan Leung Maverick Woo. 'Establishing Software Root of Trust Unconditionally.' In: *NDSS*. 2019. URL: https://www.cylab.cmu.edu/_files/pdfs/tech_reports/cmucylab18003.pdf.

[32] Serge Malenkovich. *Indestructible malware by Equation cyberspies is out there – but don't panic (yet)*. Feb. 2015. URL: https://www.kaspersky.com/blog/equation-hdd-malware/7623/.

[33] Michael Mimoso. *Release of Attack Code Raises Stakes for USB Security*. Oct. 2014. URL: https://threatpost.com/badusb-attack-code-publicly-disclosed/108663/.

[34] Michael Mimoso. *New BIOS Implant, Vulnerability Discovery Tool to Debut at CanSecWest*. Mar. 2015. URL: https://threatpost.com/new-bios-implant-vulnerability-discovery-tool-to-debut-at-cansecwest/111710/.

[35] John M. and Benjamin O. *Intel® Software Guard Extensions Tutorial Series: Part 1, Intel® SGX Foundation*. July 2016. URL: https://software.intel.com/en-us/articles/intel-software-guard-extensions-tutorial-part-1-foundation.

[36] Victor Costan and Srinivas Devadas. *Intel SGX Explained*. Cryptology ePrint Archive, Report 2016/086. 2016. URL: https://eprint.iacr.org/2016/086.

[37] Bruce Schneier and John Kelsey. 'Secure audit logs to support computer forensics'. In: *ACM Transactions on Information and System Security (TISSEC)* 2.2 (1999), pp. 159–176.

[38] G. McGraw. 'Four Software Security Findings'. In: *Computer* 49.1 (Jan. 2016), pp. 84–87. ISSN: 0018-9162. DOI: 10.1109/MC.2016.30.