Bachelor's Thesis Computing Science

Fostering Computational Thinking through Unplugged Cryptographic Activity

Martina Gaňová
s1112604

2026-06-29

First supervisor/assessor:
dr. Mara Saeli

Second assessor:
Greg Alpar, PhD

Radboud University

Abstract

This thesis explores the influence of teaching cryptography at the secondary education level on fostering computational thinking and learning graph theory. For this purpose, two 60-minute lessons were developed and taught in high school. After the lessons, students filled out learner reports. They reported gaining new knowledge about cryptography and the concept of public key encryption. They also started to think about how their data is protected online and realised that cryptography does not provide a sure guarantee. Students also increased their self-efficacy by discovering how capable they are in mathematics and computing science, and discovered that they can collaborate in mathematics. Furthermore, students mentioned multiple concepts from computational thinking, which means the lesson prompted them to engage computational thinking. We did not find any evidence that cryptography helps students in learning graph theory.

Contents

Chapter 1

Introduction

Cryptography concerns itself with keeping messages confidential by converting them into an unreadable format which can only be understood with knowledge of a certain secret (Joshi and Joshi, 2011). Cryptography is a branch of both mathematics and computing science, as it uses mathematical concepts such as graphs or matrices and combines them with computing science topics such as algorithms and computational complexity (Bartzia et al., 2024).

Cryptography is one of the most important concepts in today's world, as most of private information is stored online and needs to be protected. However, few people think about how vulnerable their data is and how hard it is to keep it safe (Bell et al., 1999). People often lack the knowledge, as no one at school ever taught them what cryptography is and why it matters. Percival et al. (2023) mentioned that, despite the fact that a middle school student could already understand a basic cryptographic protocol, cryptography is rarely introduced at this level of education. A middle school student does not need to understand a complex real-life cryptographic protocol, but they should know the basic concepts of cryptography (Percival et al., 2023). For that, protocols can be simplified to only use basic arithmetic. The complicated mathematics is needed only in real life, where the protocol is trying to protect data against attacks by computers.

Cryptography can also be used to popularise mathematics. Mathematics is often regarded as the least popular and most difficult course, especially as it is often taught mechanically (Koblitz, 1997). Teachers often show students methods of solving tasks, which students are then supposed to learn by heart and perform on request. They often do not even understand why they are doing them or how the method works. When students are asked to memorise content taught in mathematics, they miss out on the magic of discovery they could experience if only they were left to think on their own (Koblitz, 1997). Cryptography is based on mathematics, so it naturally opens the path to curiosity and discovery, as students find it fascinating how a message can be hidden (Koblitz, 1997). Students might also be motivated to try to mathematically verify the security of the protocol to see if they can safely use it. Afterwards, students can transfer this new attitude to school mathematics and discover how cryptography is based on mathematical principles (Koblitz, 1997). Cryptography can then serve as a kind of enrichment in school to popularise mathematics (Koblitz, 1997).

As topics of secrecy and hiding information are close to young people, cryptography can easily catch their attention. Most people will remember how, in primary school, they were sending messages on little papers across the classroom and trying their best to keep the contents of the message a secret. Later on, as teenagers, students also start to worry about having their own privacy, so if an activity at school is promoted as teaching them how to send secret messages, it

will naturally spike their interest (Rosamond, 2018). Many children also like the idea of imagining themselves as agents working in secrecy to stop the bad "guys", giving teaching cryptography a dramatic undertone (Fellows and Koblitz, 1992). According to Koblitz (1997), cryptography could be the activity that stands out among other courses as the exciting one where they can make discoveries and learn something practical.

Cryptography also has the potential to serve as computational thinking enrichment as it is based on both informatics and mathematics principles and can be used to create new activities that are exciting for students (Rosamond, 2018). Cryptography stands at an intersection of mathematics and informatics and relies on concepts from both (Bartzia et al., 2024). Wing (2006) has named computational thinking as one of the basic skills that everyone should have to survive in today's world. The reality of human lives in the 21st century is complex, so people should know how to simplify problems, discover concepts behind complex topics and be agile with their learning, for all of which computational thinking is needed (Wing, 2006).

Therefore, if cryptography were taught already in high school, not only would students be more aware of the risks of the online world and think about how their data is protected, but they could potentially improve their mathematical skills and develop computational thinking. Consequently, it is necessary to examine whether teaching cryptography already at secondary education can be beneficial for students.

In this thesis, we will focus on teaching public key encryption, which is a branch of cryptography that uses private and public keys for encrypting messages. In comparison to classical cryptography, the sender and receiver do not have to meet and agree on a shared secret in advance. Instead, the receiver can just publish their public key, which can be used by the sender for encrypting. The message can then be decrypted with a private key that is known only to the receiver.

Rosamond (2018) has suggested that public key encryption could also serve as an enrichment to computational thinking. Therefore, the main research question of this thesis is:

To what extent can public key encryption prompt students to engage in computational thinking?

Koblitz (1997) claimed that cryptography can be used to teach mathematical concepts naturally. In this thesis, we developed a lesson that uses graph-based cryptographic protocols to show the concept of public key encryption. Consequently, we can examine whether these protocols could be used for teaching graph theory. Therefore, an additional research question of this thesis is:

To what extent can students learn about graph theory using graph-based methods of public key encryption?

Chapter 2

Related Work

2.1 Teaching Cryptography at Schools

Cryptography is based on fundamental mathematical and computing science (CS) principles, so it can easily be turned into a vehicle for teaching those principles to children (Bartzia et al., 2024; Fellows and Koblitz, 1992). As modern-day cryptography is still coming from the same basis, once protocols used in real-life are simplified to their conceptual basis and become understandable by kids, the fundamentals will surface. An advantage of cryptography over the usual way of teaching mathematics is that cryptography places mathematics in a dramatic setting as children are trying to send secret messages to their classmates or are put in the position of agents trying to defeat the bad "guys" (Fellows and Koblitz, 1992; Koblitz, 1997). They might even relate to the topic of decrypting as they are trying to decrypt and understand the adult world, or to the secrecy, as they want to hide information from adults and have a desire for privacy (Fellows and Koblitz, 1992; Rosamond, 2018).

Mathematics is often taught mechanically in schools and, therefore, often regarded as boring compared to other subjects (Koblitz, 1997). Students are led to be passive in class by learning just a predictable, simple route which gets them the solution and gives them no space for their own contribution (Fellows and Koblitz, 1994). There have been multiple efforts to change this, such as developing whole-class discussions as a practice of teaching mathematics by Kooloos (2022). Cryptography has the potential to be another practice that will bring the joy of discovery back to students, as by participating in cryptographic activities, they might uncover and adopt new mathematics concepts in a natural way (Koblitz, 1997). Cryptographic activities can lead to mathematics being reintroduced as a culture of questions, not unchangeable answers, as it is traditionally taught (Rosamond, 2012). In the traditional way of teaching, the choice of problems is narrowed down to repetitive examples that provide immediate right/wrong gratification (Fellows and Koblitz, 1994). Cryptography activities counteract that by letting students struggle when finding a way of breaking a protocol, when there might not even exist one that is feasible for humans.

Another advantage of teaching cryptography is that it has a potential for adidacticity, meaning that students could learn independently without a teacher (Bartzia et al., 2024). Cryptography provides many inputs from the environment, for example, if children are trying to verify the security of a protocol, each of them might have a different idea for breaking it, so they need to cooperate. They are getting environmental input from the protocol by focusing on what they discovered about the secret, which motivates them to look for a different solution and provides guidance on whether they are going in the right direction (Bartzia et al., 2024). Another unusual

aspect of teaching cryptography is that it truly combines mathematics and CS together, instead of being a more common cross-curricular approach where one field is just used to explain the other (Lindmeier and Mühling, 2020).

Among students, there is sometimes an assumption that CS is just programming and that for any problem, a code can be written. Cryptography can tear down these beliefs as the security of protocols is based on computers not being able to tackle a mathematical problem in a feasible time (Rosamond, 2018).

The main concern when introducing cryptography protocols to people outside of CS is accessibility. This is an additional condition for choosing cryptographic protocols, as they also have to be secure and efficient (Fellows and Koblitz, 1992). For the public to know more about what is going on in the cryptography world, scientists must communicate in an accessible way to pass on the excitement of discoveries to people with no knowledge of their field (Rosamond, 2018).

When teaching cryptography, students should also learn about what real-life implementation looks like, how their data is protected and what makes transporting data over the internet so unsafe (Lindmeier and Mühling, 2020). With more knowledge about modern cryptographic protocols, they can make informed decisions about how they want to protect their data online (Percival et al., 2023). To increase the digital literacy of students, it is enough to just give a basic overview without trying to explain all the details of protocols that are currently deployed (Lindmeier and Mühling, 2020). Mainly, students should understand the motivation for encryption and see the risks of leaving their private data unprotected online (Lindmeier and Mühling, 2020).

2.1.1 Kid Krypto

Fellows and Koblitz (1992) have developed Kid Krypto - a series of activities meant to introduce cryptography in an interactive form to middle and high school students. They call it the crayon-technology, as one only needs a few pencils of different colours and paper to imitate real-world cryptographic protocols. Kid Krypto consists of multiple graph-based cryptographic protocols that are simple enough to be understood by children, and despite looking secure, they can be broken using some basic algebra. The protocols use properties of graphs, such as the perfect dominating set, to secretly transfer information to others.

2.2 Computational Thinking

Wing (2010) proposed to define computational thinking (CT) as:

> Computational Thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent.

Since the definition uses terms such as “problem”, it might look like CT can be applied only in informatics. However, the definition of the word “problem” is much broader, and it can stand for any real-life problem people encounter and want to solve (Wing, 2010). While looking for a solution, they can, thanks to CT, employ different strategies from CS, such as reducing, transforming or embedding (Bell et al., 2009; Wing, 2006). For example, imagine a person who can communicate only via blinking. The basic approach to communicate with them would be to spell out the whole alphabet and wait for a blink. But someone who uses CT might come up with a more efficient solution to split up the alphabet using the concepts from the divide and conquer algorithm (Bell et al., 2009).

Another necessary part of the definition is the information-processing agent, which highlights the difference between thinking like a computing scientist and like a mathematician. While mathematicians are focused on finding a solution, or an equation defining an answer, computing scientists would like to see the process of how the problem was solved (Nardelli, 2019). In CT and, therefore, also in CS, the result is a process that solves the problem and can be understood by the information processing agent. Aho (2012) even scratched the information processing agent from the definition and instead defined the solutions to be in the form of "computational steps and algorithms" (p. 1) to point out that in CT, the solution is not an answer to a problem, but rather a set of instructions that solves it.

Abstraction is one of the most important concepts in CT, as it can be used to have one object represent many (Wing, 2010). With the use of CT, people are able to group objects by their essential properties while overlooking irrelevant ones, similarly to how a programmer would create an abstract data type to represent different concrete objects which share certain properties (Wing, 2006). In programming, abstract data types are the key to scaling as they can be used to represent unforeseen objects and simplify the system by shifting the focus on individual layers (Wing, 2010). Abstraction removes the unnecessary information from a problem and brings forward the concepts relevant to solving it (Grover and Pea, 2013).

CT is based on mathematical concepts, as also CS is based on mathematics. CS was developed when mathematicians, who instead of trying to solve problems, moved on to attempting to automate the creation of proofs and offloading the actual problem-solving part onto machines (Nardelli, 2019). In CT, in contrast to mathematics, the domain of problems and solutions is limited just by human inventiveness and curiosity, as there are no formal constraints on expressing the problem in exact numbers or only allowing a limited domain of solutions (Wing, 2006, 2010). The human mind has the skill to imagine any problem and adapt an already known strategy to solve it using CT (Wing, 2010). Therefore, CT is not trying to teach people to think like a computer, but rather to teach them how to use the same strategies as computers to solve problems (Bell et al., 2009; Wing, 2006).

CT should become one of the basic skills, such as reading, writing or arithmetic, taught at primary and secondary schools, as everyone, not just computing scientists, can benefit from it (Wing, 2006; Katai et al., 2014). Even though CT does have an overlap with logical thinking and systems thinking, it brings a lot more to the table, such as pattern matching or recursive thinking (Wing, 2010). Wing (2010) even calls CT the "new literacy of the 21st Century" (p. 3), as in the current age, problems are coming from all sides, and people need to have the skills such as CT to efficiently deal with them. CT is unique in that it focuses on principles and methods instead of systems and tools (Nardelli, 2019). In the centre, there is an attempt to think like a computing scientist and to move beyond specific problems to seeing the core concepts of CS (Nardelli, 2019).

To further promote students' interest in informatics and develop their CT, a challenge-contest named Bebras was started in Lithuania (Dagiene and Stupuriene, 2016). Even though the competition started with an idea to introduce CS in an unusual way and attract more students to the field, it has also created a large corpus of tasks meant to test CT (Calcagni et al., 2017). As they are easy to access and well signposted on the Bebras website, they are often reused for other purposes, such as research (Calcagni et al., 2017).

2.3 Unplugged Computer Science

Students often make the assumption that CS is just programming and only people who do it are antisocial nerds, so students have little interest in learning more about it (Dagiene and

Stupuriene, 2016). Not only do they then miss out on the opportunity of finding a potential career, but they also have a narrow-minded point of view on CS, which leads them to lack all the other computing skills, such as working in spreadsheets, which is useful in many other fields (Dagiene and Stupuriene, 2016; Bell et al., 2009). This opens a need for CS activities that do not necessarily involve interaction with a computer and can therefore spark students' interest in an easier way (Bell et al., 2009).

Due to these concerns, an initiative "Computer Science Unplugged" was created (Bell et al., 2009). On their website, csunplugged.org, they present activities meant to introduce CS concepts without using a computer. Once the computer is taken out of the equation, students can focus purely on the concepts in the activity and get an extra motivation to complete it, as each activity is presented as a story (Bell et al., 2009). For example, in one activity, they are searching for a treasure on a pirate map, which in reality is just an automaton that they would normally immediately write off as a boring topic. In another activity focused on sorting algorithms, they might have to imitate sorting the way computers do it by only being allowed to weigh at most two items on the scale at a time. This presents a kinesthetic implementation of how a computer works, so students can try sorting objects like a computer without even having to touch a keyboard (Bell and Lodi, 2019). Fellows and Koblitz (1992) as authors of Kid Krypto activities also argue that teaching cryptography is best done with the unplugged approach, as it teaches children how to work with algorithms and develops their logical modes of thought in a creative and exciting way. Activities using the unplugged approach are focused on letting students discover the algorithms themselves by solving a problem located within a constrained world instead of spelling out the algorithm (Bell and Lodi, 2019). The constructivist nature of the activities brings students a feeling of empowerment after they discover a solution to a difficult problem (Bell and Lodi, 2019).

The unplugged approach also makes it possible to teach about CS in places where schools do not have access to computers (Fellows, 1991). Students can then discover CS concepts and decide if they would like to study it more later, no matter what kind of primary school they went to. Even if students receive just 1 hour of an unplugged activity, it can already have a lot of influence (Bell et al., 2009). That makes it easier to implement than other popularisation activities for CS, such as mentoring or teaching programming, which require much more time to be effective (Bell et al., 2009).

Unplugged approach inverts the traditional order of first teaching the basics of programming and only afterwards the more advanced concepts, such as different algorithms (Bell and Lodi, 2019). It is not meant to remove programming completely, but rather to first start by understanding the big ideas of CS and only then move on to learning how to program. Hermans and Aivaloglou (2017) found that children who first begin learning to program with an unplugged approach and only later move to programming on computers show greater self-efficacy and are more willing to explore and use unknown commands than children who were using computers from the start. In unplugged activities, students can immediately dive in and try out the algorithms, which helps to develop not just their problem-solving skills, but can also help those who later decide to continue in a career as a computing scientist (Bell and Lodi, 2019).

Unplugged activities work best if they are followed by plugged activities so students can apply what they have learned and also get more immediate feedback from the computer than if it were from a teacher (Bell and Lodi, 2019). Taub et al. (2012) found that even though students understood the nature of CS more after taking part in unplugged activities, their interest in studying CS decreased. The authors assumed that, as students did not see the connection to how actual computers work or what it would look like if they went on to study CS later, the idea of becoming a computing scientist still seemed abstract to them. Thies and Vahrenhold (2013) on the other hand found that in lower secondary education, if the CS unplugged curriculum is woven into the regular one, it has at least as good results as the traditional curriculum.

The teacher in unplugged methods (similarly to other pedagogical approaches, e.g. problem-based learning) also needs to take on a different role and go from acting as the bearer of knowledge in the classroom to becoming a facilitator who is there to help students experiment with their own creativity (Bell and Lodi, 2019). A teacher might have a certain path on how to get to a solution in mind, but as the students are the ones coming up with the ideas, the knowledge ends up being constructed by them (Bell and Lodi, 2019). Also, the instruction is much shorter as it usually consists just of one or two sentences, after which students can get into building the knowledge (Bell and Lodi, 2019). As new curricula for CS are filled with scientific words with abstract meaning, many teachers feel intimidated by having to explain them to their students (Bell and Lodi, 2019). The unplugged method can help with that as the teacher can first do an activity with students, and after they understand the concept, the teacher can give them the terminology of what they did (Bell and Lodi, 2019).

Feaster et al. (2011) has found limitations in trying the unplugged approach with high school students, as they seem less interested in doing kinesthetic activities than primary or middle school students. They also mentioned that these students are more likely to have already had some encounters with CS and have a preference to keep learning the way they started - with computers - while the younger students usually had their first encounter with CS through the unplugged methods, so they accepted that as the default way of learning.

Chapter 3

Instructional Design of the Learning Activity

The learning activity was aimed at high school students, as they already have enough mathematical knowledge to not just understand how the protocol works, but also to potentially break it (Bell et al., 2015). Even though the unplugged approach was found to be most effective with middle school students (Feaster et al., 2011), we chose to do the activity with high school students to increase the chance that students would have the skills to break the protocol. We adapted the unplugged approach to not use fairytales in the activities and rather focused on presenting the lesson as teaching about concepts which are used in real-life cryptography, with the hope that it would pique the interest of high school students, considering they often transfer their private data over the internet.

The activity focused on public key encryption, as it is nowadays widely used in real life. Students are intrigued by this, for them unusual application of one-way functions and find it fascinating that a function can be easy to compute one way while the other way is close to impossible (Keller et al., 2010).

3.1 Lesson Design

3.1.1 Motivation behind the Lesson Design

The lesson began by asking students a few warm-up questions to capture their interest and establish their current knowledge of cryptography. As cryptography is not part of the usual curriculum, it was necessary to establish what students already know from sources outside of school. During the lesson, simplified terminology was used, such as graphs being called maps, the private key being referred to as the secret map, and the adversary being referred to as the hacker, as these terms are more familiar to students. The warm-up questions at the beginning of the lesson were designed to encourage students to speak up and spark their interest by presenting the lesson as learning the basic principles of how encryption on the internet works. They framed the lesson as learning both how to send secret messages and how to act like the adversary. Answers to the more knowledge-based questions helped with introducing the concept of asymmetric cryptography, as it was presented in relation to the kinds of encryption students already knew.

Next, the basic concept of public key encryption was introduced with the metaphor of a paper dictionary that only provides one-way translation from English to an exotic language

(Sústrik, 2013 in Rosamond, 2018). Students could then see the most basic principle of public key encryption in an intuitive real-life example without having to understand complicated protocols. The metaphor was also used in the following parts of the lesson when introducing the graph-based encryption protocol to help students connect the concept with the realisation.

In the main part of the lesson, two graph-based encryption protocols were introduced. The first one was the original encryption method used in Kid Krypto (Bell et al., 1998; Rosamond, 2018; Fellows and Koblitz, 1992) that uses a perfect code dominating set in graphs. Rosamond (2018) defines dominating set as "a subset S of the vertices such that every vertex in the graph is either in S or is adjacent to at least one member of S. The vertices of the graph are said to be *dominated* by the vertices of S." (p. 98). A perfect code is a dominating set where each vertex is dominated by exactly one vertex (Rosamond, 2018).

In the protocol made by Fellows and Koblitz (1992), a number can be sent by writing numbers on each vertex, such that their sum is the secret message. Afterwards, the numbers on the vertices are rewritten to be equal to the sum of the numbers on the vertex itself and on all of its neighbouring vertices, and the map can be sent to the receiver. The receiver uses the perfect code dominating set as the private key and can obtain the message by adding up the numbers on all the dominating vertices.

When trying to break the algorithm in the lesson, the graph (3.1, 3.2) proposed by Keller et al. (2010) was used, as it is simple enough that, with the right idea, students can carry out the attack rather fast. For trying to find the secret key from the public one, we created a new graph (3.3).

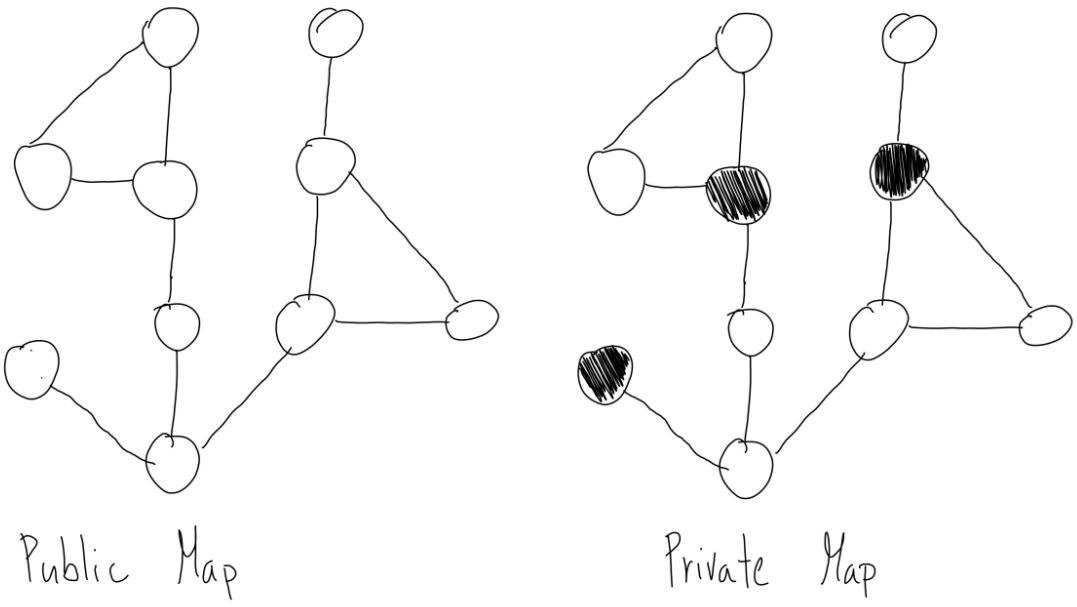

Figure 3.1: Public and private map

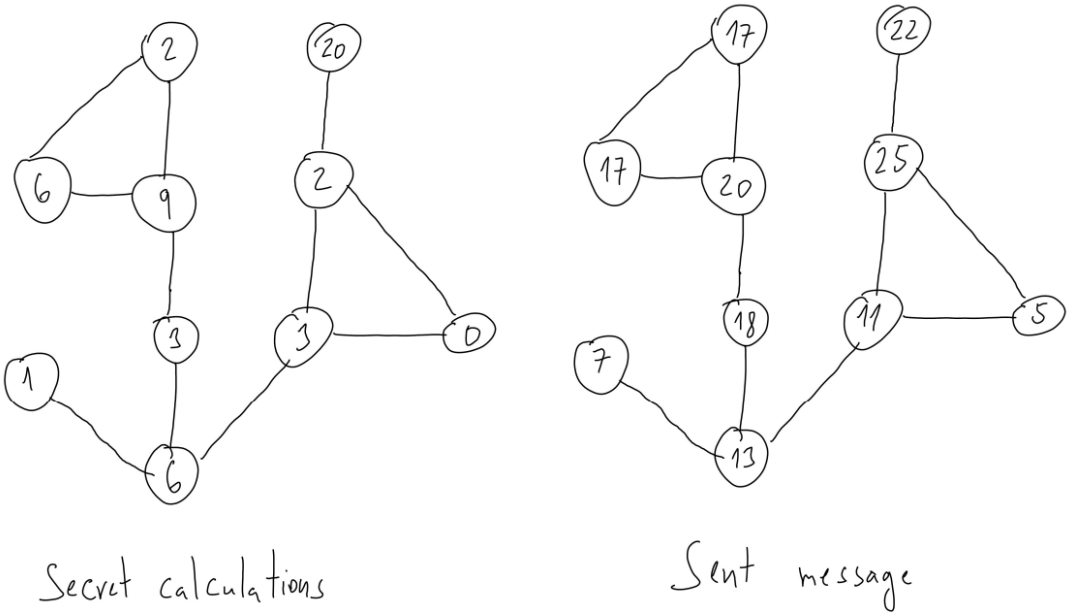

Figure 3.2: Example of calculations of encryption

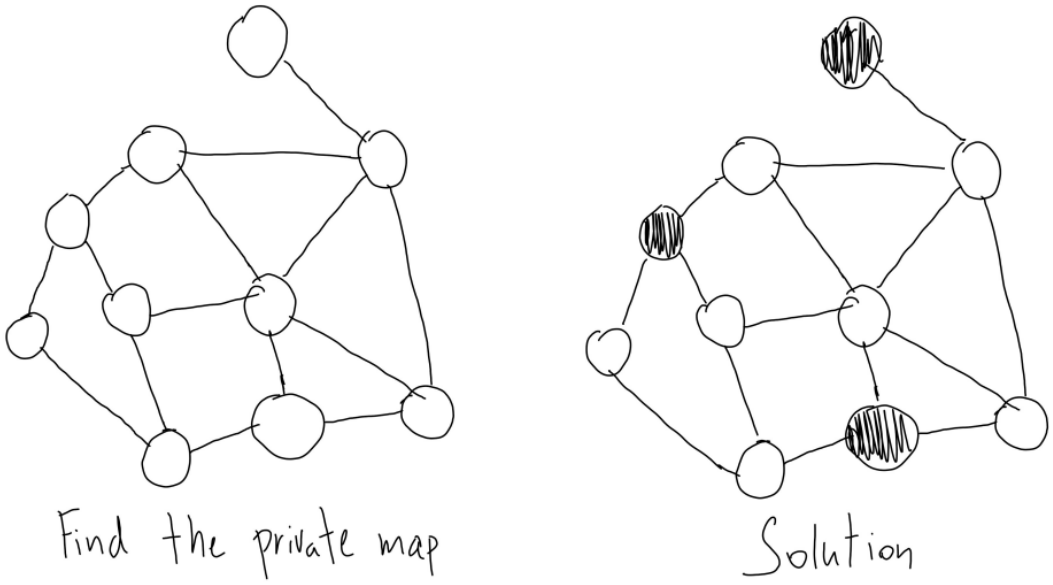

Figure 3.3: Graph for finding the private key

The second encryption method was developed by Rosamond (2018) and uses the disjoint

directed cycle cover of a graph. A directed graph is a graph where each edge also has a direction. A directed cycle is a cycle of vertices that ends and starts at the same vertex. A cycle cover means that all vertices of the graph are included in the cycles. A disjoint directed cycle cover means that the cover is made up of one or more directed cycles that do not have any vertices in common, and together they include all vertices of the graph Rosamond (2018).

Using the protocol developed by Rosamond (2018), a number can be sent as a secret message by writing a number on each vertex such that their sum is the secret message. Then on each edge, a number is written that is equal to twice the number at the beginning of the edge minus the number at the end of the edge. The numbers on the vertices are then erased, and the map can be sent to the receiver. The receiver can read the message by adding up numbers on all the edges that are in a disjoint directed cycle cover. The disjoint directed cycle cover of the graph then serves as the private key.

For breaking the protocol, the original graph (3.4, 3.5) suggested by Rosamond (2018) was used. For finding the private key from the public one, we created a new graph (3.6).

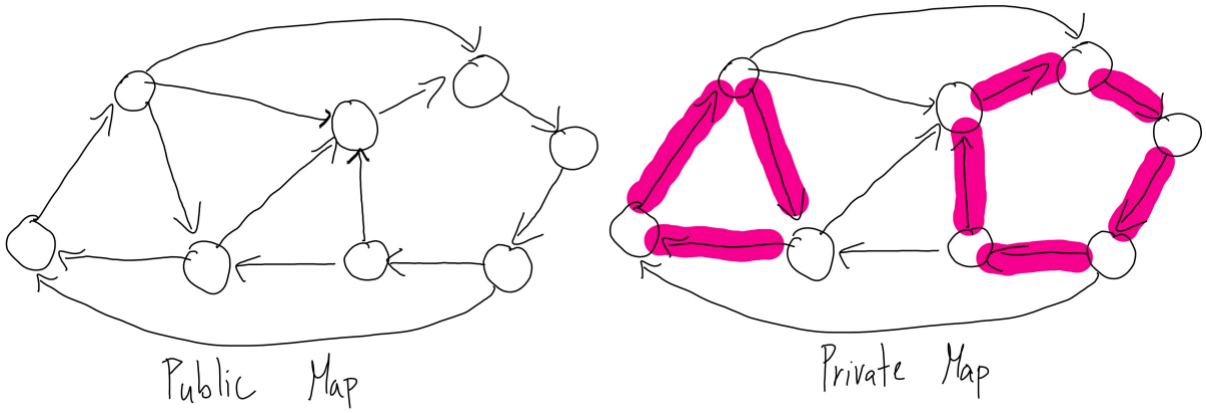

Figure 3.4: Public and private map

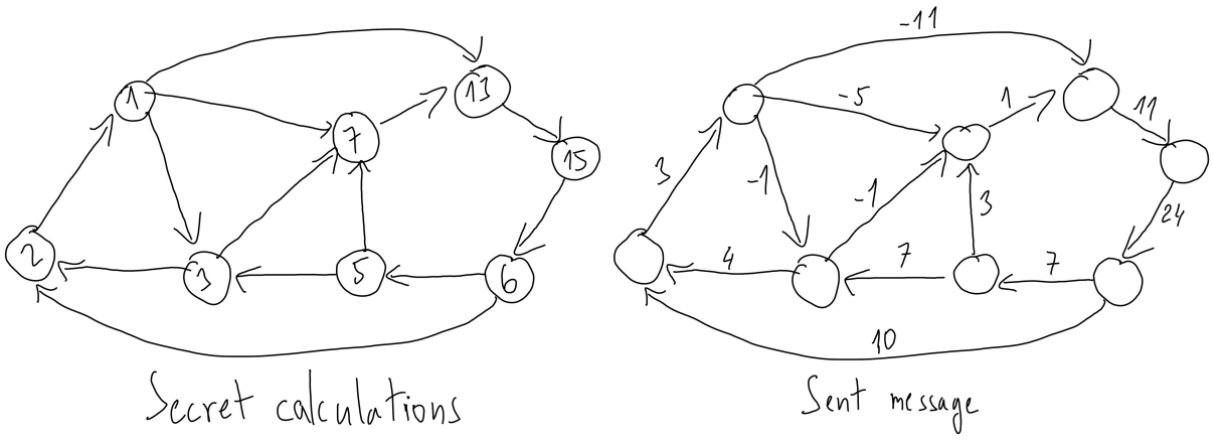

Figure 3.5: Example of calculations of encryption

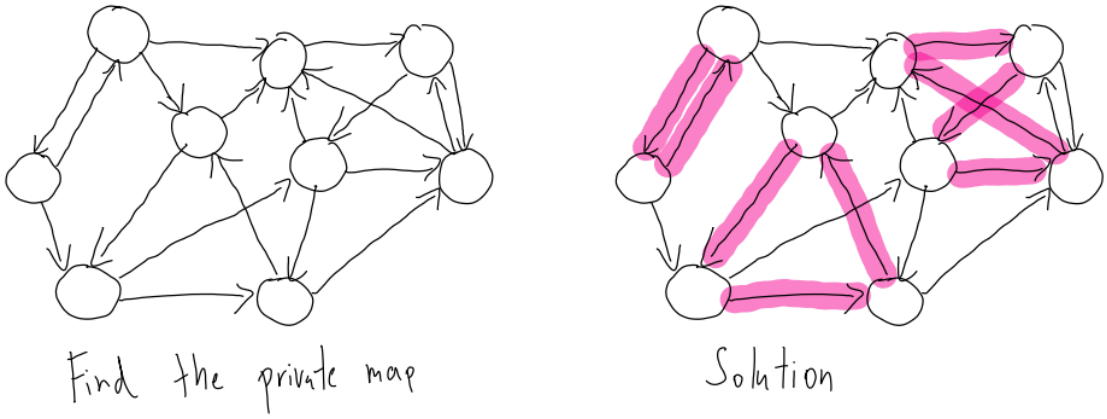

Figure 3.6: Graph for finding the private key

The scheme of the lesson was the same for both protocols. However, we expected that students would break the second protocol faster as they already had knowledge from breaking the first cryptosystem. Following the lesson design by Keller et al. (2010), students first tried encrypting a message. Afterwards, they took on the role of an adversary and competed against the teacher with a private key on who could find the secret message faster. We expected students to figure out a way to find the secret using linear algebra, which would take them significantly longer than for the teacher with a private key, who only needs to do a few additions. As it was likely that students would lose the contest, they became interested in what the private key looks like and why it works faster than the adversary's way of calculating the result.

Afterwards, students tried to figure out why the private key works and tried to develop their own private and public key pairs to familiarise themselves with the protocol further. At the end, students were asked multiple questions to guide them to think about how the private key is created and why it is easy to find the public key from the private one, but difficult the other way

around. This helped them to further establish how the basic concept of public key encryption is applied in this protocol and to what extent it is secure against adversaries.

By then, students seemed to be confident in understanding how the graph-based encryption works and which properties of the graph it uses. Therefore, in the next part of the lesson, they got a task that required them to transfer the knowledge about the graphs in cryptographic protocols to a different context. The task could be solved using the same principle as if an adversary were trying to find the private key from the public one, which is feasible to do by hand in smaller graphs (Rosamond, 2018). Students could apply the same principles they used when trying to create their own set of private and public keys. For the perfect dominating set cryptosystem, the used graph (3.7) comes from the original Kid Krypto activity by Bell et al. (1998). It was chosen as the solution is not obvious right away, and students have to experiment a little until they discover it. For the disjoint cyclic cover cryptosystem, we created a new graph (3.8) that could be given as a verbal task and is a lot easier to solve once written down in the form of a graph.

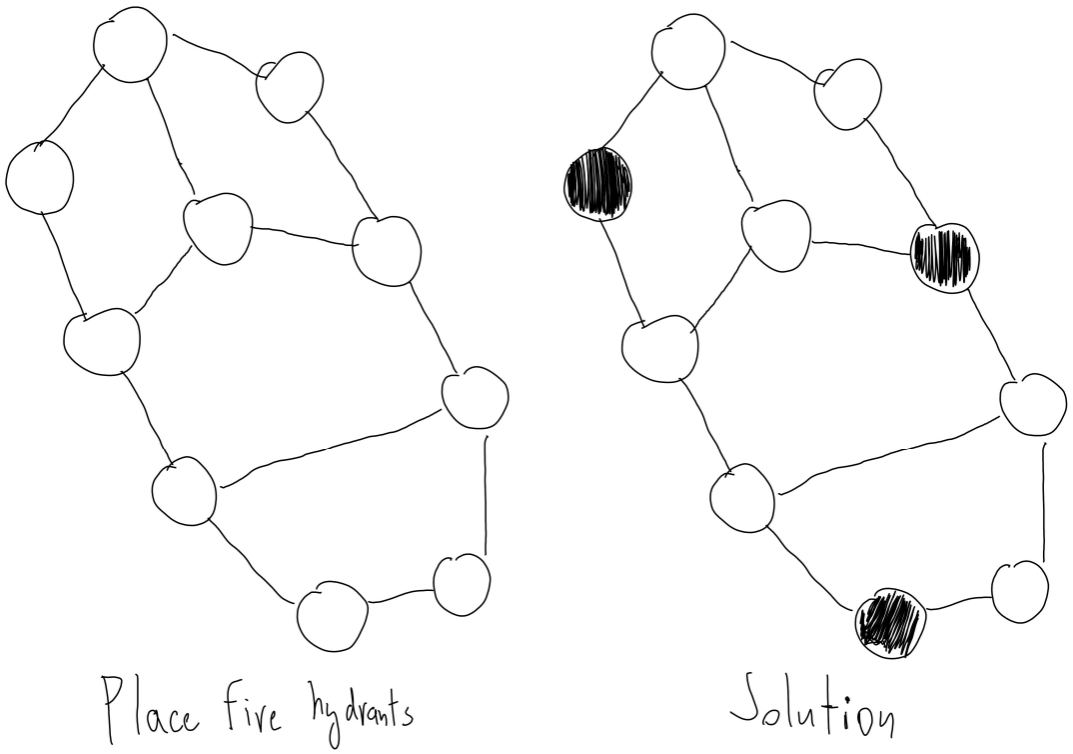

Figure 3.7: Graph for transfer of knowledge

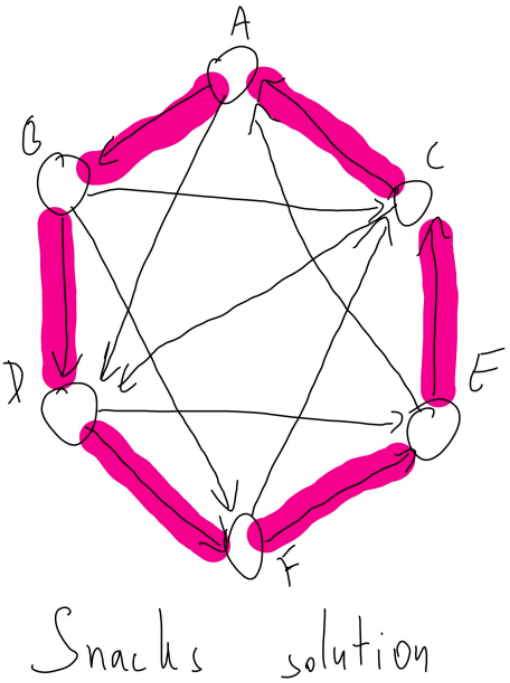

Figure 3.8: Graph for transfer of knowledge

Percival et al. (2023) emphasised that it is necessary to teach students also about modern cryptographic protocols that are still in use. Therefore, the goal of the last part of the lesson was to connect the knowledge students just gained to its real-life applications. For that, the Diffie-Hellman-Merkle (Diffie and Hellman, 1976) protocol was chosen as its execution is simple enough so that even high school students can understand it, and the protocol is still in use today. It is also a good example of how asymmetric encryption is often used in combination with symmetric encryption, so students get an idea of how protocols are usually used.

3.1.2 Learning Objectives

The activity was designed with these learning objectives in mind:

- students should understand the basic principle of public key encryption - know the difference between private and public key and see the advantage of asymmetric cryptography compared to symmetric, realise that finding a public key from a private one is much easier than the other way around
- students should learn to think like an adversary and be able to look for ways to break protocols on their own
- students should be prompted to think about how their data is protected by cryptography and realise that cryptographic protocols cannot be completely secure
- students should see that the mathematical properties they are learning in high school can also be applied in real-life to provide security to cryptographic protocols, such as the application of powers and logarithms used in Diffie-Hellman-Merkle

3.1.3 Pilot Run

Before implementing the activity at school, there was a pilot run with 14-15-year-olds at a camp for mathematically talented children in Slovakia. As the learning activity was planned for 2 hours, but at the pilot run, we only had 1 hour of allocated time, there was time for only one of the graph-based protocols. We decided to do the perfect dominating set protocol as it is easier to understand than the one using a disjoint directed cycle cover.

The warm-up questions proved to be a good starting point for the teacher to establish students' knowledge and helped connect the newly gained knowledge to what students already knew later in the lesson.

Originally, the lesson started with introducing the concept of public key encryption through a simple addition protocol with private and public numbers. However, at the pilot run, it became clear that adding numbers and arithmetic operations into the protocol so early in the lesson resulted in confusion and the concept of public key encryption was lost in between all the arithmetic operations. Therefore, this part was replaced with the dictionary metaphor.

The encryption by graphs was easy to understand for students, and they were also able to break the protocol without much struggle. Each time a group cracked a message, they got a boost in energy as the feeling of finding someone else's secret was exciting for them. Each group progressed at a different speed, which meant that the following parts of the lesson were done individually per group. When implementing the lesson, the teacher should have more extra activities ready for the faster group, to even out the pace and allow for reconvenement with the whole class for discussion. Students were also easily able to understand how the private key works and ended the lesson by creating their own public and private key pairs, which they were planning on asking other camp participants to send them messages with.

In the pilot run, the first part of introducing the basic concepts took much longer than expected, which meant that at the end of the lesson, there was no time for the transfer of knowledge part. For the actual lesson, switching to the dictionary metaphor helped with that.

3.2 Finalized Activity Design

In this section, we present the activity design. It was scheduled to last for two 60-minute lessons. The activity began with warm-up questions, followed by a metaphor for introducing the concept of public key cryptography, after which two graph-based encryption protocols were introduced. For each protocol, students tried encrypting and decrypting the message and being an attacker. Afterwards, there was a short activity to try transfer of knowledge to graphs and the lesson finished with showing Diffie-Hellman-Merkle protocol as an example of real-life cryptography.

Warm-up questions (5 min)

Do you use the internet a lot? Have you ever thought about how your data is protected on the internet?

Have you ever tried to hack anything to get information you should not have access to?

Do you know any methods for sending encrypted messages that were used in the past, even before the internet existed?

Do you know any methods of encryption that are still in use today?

Have you ever heard the term cryptography? Do you know what it means?

Introducing public key encryption (5 min)

Have you ever seen a paper dictionary? It is a very long book that has the translation from one language to another, but not the other way around. Imagine you were the only person in the

world who owned a dictionary from the Inuit language to English. You could then give everyone the dictionary from English to Inuit language. People would now be able to send you messages in Inuit language that no one other than you could read.

Questions

Why can't anyone else understand the messages?

How could someone else try to recover the message just with the English to Inuit language dictionary?

What is the advantage of this kind of system compared to, e.g., the Caesar cipher? (or some other method that was mentioned in the warm-up) Hint: agreeing on the key

Graph-based public encryption activity based on perfect dominating set (50 min)

Now we will try using a more complicated system for creating messages. It works by having a public map that is used for encryption and a private map for decryption. With the map, you can send a single number as a message. First, you write a number at each intersection such that their sum is equal to the secret message. Next, you write on a copy of the map at each intersection the sum of all its adjacent intersections and itself. Then you keep the first map secret, and the second one you send. (emphasise that all this information is public)

I have created a public and private map for myself, so that you can send me messages. We will split into groups of three, and each group will try sending me a single number as a message. Then we will have a little contest on who can decrypt the message faster. Each group will pass a copy of their map to another group. The other group will then take on the role of a hacker and will try to decrypt the message without having the secret map. Meanwhile, I will try to decrypt messages from all the groups faster than you, as hackers will. (20 min)

(whenever a group decrypts the message) This is the private map; it works by having some intersection highlighted. Once you write down numbers at all the highlighted intersections, you get the secret message. Try to verify if the message you decrypted is the same as the one you get with the secret map. How does the secret map work? (5 min)

Given this simple public map, can you derive the private map from it? Extra: remove one intersection and try to find a private map. Is it always possible to find one? (5 min)

Now, try developing your own private and public map in the group, so that other groups can try sending you messages. Whoever has it ready can give their public map to another group and ask them to send a message. (10 min)

Discussion (10 min)

How difficult was it to encrypt the message? And to decrypt it without having the secret map?

How difficult was decrypting for you as a group compared to the teacher with the secret map?

Which map is represented by which dictionary from the Inuit language to English example?

How do you think this protocol works? Why is it enough to sum just numbers on highlighted intersections in the secret map to obtain the message? (Hint: Which intersections are added up to get the number at the highlighted intersection?)

How is the public and secret map created? When creating your own, how did you start? (emphasise that going from public map to private is more difficult)

How could you send an actual message made out of letters with this system? (Hint: mod 26)

Break between lessons

Graph-based public encryption activity based on disjoint cyclic cover (25 min)

Let's now look at a different protocol that uses a map with just one-way streets. Again, there is a public map that will be given out to everyone. A message can be encrypted by writing a number

at each intersection such that their sum is the secret message. On the copy of the map, write on each street the sum of twice the number at the beginning of the street minus the number at the end of the street. The first map you should keep secret, and the second one you can send to the other group.

We will again have a contest where I will try to decrypt all the messages with the secret map faster than you, as hackers will. (10 min)

(whenever a group decrypts the message) This is the private map; it works by having some edges highlighted. Once you add numbers to all the highlighted edges, you get the secret message. Try to verify if the message you decrypted is the same as the one you get with the secret map. How does the secret map work? (5 min)

Extra: Given this simple public map, can you derive the private map from it? Now, remove one intersection and try to find a private map. Is it always possible to find one?

Now, try developing your own private and public map in the group. Extra: Whoever has it ready can give their public map to another group and ask them to send a message. (5 min)

Discussion (5 min)

How difficult was it to encrypt the message? And to decrypt it without having the secret map?

How do you think this protocol works? Why is it enough to sum just numbers on highlighted streets in the secret map to obtain the message? What is interesting about the arrangement of the highlighted edges on the secret map?

How did you start when you were trying to create your own secret and public map?

Transfer of knowledge (10 min)

The mayor from a nearby city came to us for help as they heard that we are very skilled with maps. The mayor needs to dedicate some houses to be emergency fire houses that will have a hydrant in front of them. The mayor would like each house to have a hydrant located at a distance of at most 1 street. To avoid confusion when the firefighters come, the mayor wants exactly one hydrant to be at that distance, so they will know where to go right away. Can you help the mayor find where should the hydrants be placed?

Students came to us with a problem about their snacks, as many of them dislike the snack they have and want to switch it with someone else's. They have heard that, as we understand the encryption system so well, we should also be able to solve their problem.

The students brought different snacks, but no one was satisfied with theirs. How can they swap them so that everyone is satisfied? Alice wants either Bob's or Denis's snack. Bob wants either Charlie's, Denis's or Frank's. Charlie wants Alice's or Denis's snack. Denis wants Eve's or Frank's. Eve wants Charlie's or Alice's. Frank wants Charlie's or Eve's. Can you find out how they should switch the snacks so that everyone is satisfied? Try creating a map similar to the one we used for sending secret messages.

Real-life implementation of cryptography (15 min) Let's now look at a cryptographic protocol that is actually used by computers. The first-ever public key cryptography protocol is called Diffie-Hellman-Merkle key exchange, and it is still in use today. In the protocol, both parties agree on a generator G and a prime number p. They both choose their private keys and calculate their public key as $A = G^a \mod p$. Next, they exchange their public keys and calculate the shared key as $K_{a,b} = B^a \mod p$.

Let's try it out by calculating a public key for me together. Now each group can create their own public key and find the shared key that they have with me. (10 min)

Discussion (5 min)

How come we end up with the same shared key?

Why do we derive the public key from the private key and not the other way around?

How could someone try to decrypt our conversation? Should they focus on finding the shared or the private key?

How can they use this method of encryption to secure their further conversation?

Wrap up

Diffie-Hellman-Merkle is a protocol that is still in use today, even though a more common version of this protocol uses points on an elliptic curve and not powers of numbers.

In the lesson, we discovered the basic principles of public key cryptography. If you go on to study computing science, you will come across them in more detail. But even if you don't, everyone must know how cryptography can help us protect our data on the internet, but also that it can not guarantee complete security.

Learner reports (10 min)

Please fill in these learner reports.

Chapter 4

Methodology

4.1 Participants

Participants were 12 students in grades 10 and 11 (ages 15-17) at an international school which follows the IB programme in the Netherlands. These students were taking mathematics at the highest level in their grade. The two cryptography lessons took place during these students' mathematical lessons on the same day, with a one-hour break in between. Each lesson lasted 60 minutes and was taught in English. Permission from the students (and parents for students younger than 16) was obtained before carrying out the intervention and data collection. During the lessons, students worked in the same groups of 3, which were decided by the teacher and each included one student from grade 11 and two students from grade 10.

4.2 Data Collection

At the end of the second lesson, students were asked to anonymously fill in learner reports based on the format proposed by De Groot (1980). Learner reports are designed to trace educational objectives (Van Kesteren, 1993). As students fill in those learner reports themselves with what they have learned in the lesson, we can examine whether the lesson has met its educational objectives. We will try to answer our research questions by examining the students' self-reported learning.

The learner report consists of four sentences that guide students to think about what they have learned. In the report, the first two sentences regard students' learning about the world and the last two learning about themselves. The first and third sentences guide students to think about the rules and generalities, while the second and fourth are concerned with exceptions or surprises they take away (Van Kesteren, 1993).

Learner reports filled in by students consisted of these four beginning of sentences:

From the lessons about cryptography, I have learned that ...
From the lessons about cryptography, I have learned that it is not true that ...
From the lessons about cryptography, I have learned that I ...
From the lessons about cryptography, I have learned that it is not true that I ...

4.3 Data Analysis

Data from the learner reports were qualitatively analysed using inductive coding. First, the paper learner reports were digitised. Then we repeatedly read them to familiarise ourselves with the data and to discover recurring themes. Six themes emerged from data: learning about cryptography, learning about mathematics, pattern finding, interest in CS, self-efficacy (belief in their own capability (Bandura and Wessels, 1997)) and collaboration.

Quotes in each of these themes were then examined to find different perspectives in the data. We found 10 perspectives, with cryptography having 4 perspectives, mathematics having 2, and other themes each having one. Finally, we formulated learning outcomes of the lesson from the perspectives.

For example, a quote "I learned that if you put a lot of effort through a long period of time I can crack any code" (Student C) was first categorised as learning about cryptography, as the student mentions that protocols can be broken. It was further categorised as (in)security of cryptography, as the student has learned that protocols do not completely protect the data, but instead just prolong the time and increase the effort the attacker has to put in to obtain the secret.

Chapter 5

Results

5.1 Quantitative Results

We first looked at the distribution of how many times each theme was mentioned by different students. We also examined how many times the themes were mentioned by students in total. The results are shown in table 5.2.

We can see that cryptography was the most occurring theme with mentions from all students. It is followed by self-efficacy, which was mentioned by 8 out of 12 students. Remaining themes were all mentioned by 3 to 5 students. We can also see that all students mentioned at least 2 different themes in their learner reports. No student has mentioned all 6 themes, but there were 2 students who mentioned 5 different themes, with both of them missing interest in CS.

5.2 Qualitative Results

In the learner reports, students mentioned **learning about cryptography**. They reported learning about **security** of cryptography and realising that all protocols can be broken, but it usually requires a lot of time and effort. They also thought about what consequences that has for their private data. Students learned that the **purpose** of cryptography is to secure their data on the internet. Furthermore, students discovered that cryptography is **not that difficult** to understand. Another reported learning was about the **concept of public key cryptography**, such as noticing that decryption with a private key is easier than with the public one and that private keys are made before public ones.

Additionally, students reported learning about **mathematics**. They realised that mathematics has **applications in real-life** such as protecting data using cryptography or the fire hydrant problem. Three students pointed out the use of logic in cryptography, and one of them also mentioned using algebra. Interestingly, there was a student who viewed logic as a separate field from mathematics. Moreover, the lessons led students to realise that mathematics is **more diverse** than they thought before. They saw different patterns and methods that appear in mathematics and can be used for the purposes of cryptography. One student even mentioned that cryptography is an area of mathematics they didn't know about before.

In the learner reports, students often mentioned discoveries about **pattern finding**. Two students mentioned learning about their own ability to find patterns. Some students noticed how finding patterns or discovering sequences was helpful for figuring out how to find the secret message. Student F pointed out that "even complex patterns can be understood in simpler

Student	crypto-graphy	mathe-matics	pattern finding	interest in CS	self-efficacy	collabo-ration	unique themes
A	2	2	0	0	2	0	3
B	2	1	2	0	2	1	5
C	2	0	2	0	0	0	2
D	1	0	0	0	1	0	2
E	2	3	1	0	1	0	4
F	1	2	1	0	1	1	5
G	1	0	0	2	0	0	2
H	2	0	0	1	1	0	3
I	2	0	0	2	0	0	2
J	2	0	0	0	2	1	3
K	1	0	0	0	1	2	3
L	1	1	0	0	0	0	2
unique students	12	5	4	3	8	4	
total mentions	19	9	6	5	11	5	

Table 5.2: Distribution of students' quotes in themes

ways".

Another recurring theme was **interest in CS**. Two students were surprised to find that they liked cryptography or data science. Two students learned that they find this topic interesting, and another student reported discovering that they are interested in CS and algorithms.

Many students mentioned how their belief about their own skills (**self-efficacy**) was influenced by the lesson. Multiple students expressed surprise about being able to understand cryptography; one student stressed it even further that they discovered they can understand a new topic in just 2 hours. Furthermore, one student learned from the lesson that they are capable of going on to become a computing scientist, and another student realised that they can become a hacker. There was also a student who realised that they could solve problems in the lesson using just the skills they already had. Student B has reported learning that they can "persist when facing challenges".

Students reported learning about how helpful and pleasant **collaboration** in a group can be. One student was surprised that even in mathematics, it is possible to cooperate with others. Student K has noted that they learned that they "can sometimes listen to the other's opinion instead of calculating everything [themselves]".

Quotes for each category and its subcategories are shown in 5.3.

Table 5.3: Quotes from different categories

Categories	Perspectives
Cryptography	**Difficulty of cryptography**
	• [I have learned that it is not true that] cryptography is hard. (B)
	• Basic cryptography is not that difficult. (J)
	• It is not true that basic cryptography is difficult. (K)
	(In)security of cryptography
	• [I have learned that it is not true that] all my information is completely secure and that although it was simple, it is always possible to obtain the private key. Even completely secure encryptions like cryptocurrencies are never guaranteed. (A)
	• [I have learned that it is not true that] encryption is easy to crack, that everyone has access to your information. (C)
	• That hacking the code is far harder than making it. (E)
	• [I have learned that it is not true that] all cryptographic patterns and messages are extremely sophisticated and impossible to break. (F)
	• Decryption can be nearly impossible to brute-force, especially when complicated encryption methods are used. (H)
	• I learned that if you put a lot of effort through a long period of time I can crack any code. (C)
	Purpose of cryptography
	• [patterns in mathematics] are used to protect private data on the internet. (E)
	• Cryptography is about securing data. (G)
	• Cryptography isn't about languages necessarily, but is about securing data. (H)
	• [I have learned that it is not true that] cryptography is only about languages, but more about data security. (I)
	Concept of public key encryption
	• I have learned that to decipher an encryption you have a public key and a private key: [Inuit language] $->$ English, where it is more secure to go from private to public. This was supported by the way to convert public to private, which involves calculating sum of areas where there are most connections to quickly get the private key. (A)
	• public and private keys (B)

Continued on next page

Categories	Perspectives
Cryptography	• [I have learned that it is not true that] public keys are made first, rather private key would be easier. (D) • Cryptography is about securing data using certain formulas to encrypt and decrypt. (G) • To decrypt a message, we need to use the arrows to make closed systems only going through each circle once. (I) • Also, there are private and public keys. (J) • There is private map that makes decrypting easier. (L)
Mathematics	**Applications of mathematics** • It is very possible to apply different methods or mathematical passages to new areas of the subject. (A) • I learned that cryptography includes an interesting field that applies math to encrypt and decode codes. (B) • Interesting complex patterns exist in mathematics and they are used to protect private data on the internet. (E) • even logic can be used. (F) • [I have learned that it is not true that] only math is needed to solve. (L) • Only math must be used to solve real-life problems; logic may also be used (such as the fire hydrant problem) (E) • [I] can use logic and algebra to solve/decrypt these kinds of messages quite easily. (F) **Diversity of mathematics** • It is very possible to apply different methods or mathematical passages to new areas of the subject. (A) • Interesting complex patterns exist in mathematics (E) • There are many areas [in mathematics], like cryptography which I know very little about. (A)
Pattern finding	**Pattern finding** • It focused a lot on pattern finding and finding ways to isolate numbers. (B) • Codes follow a sequence and once you understand it is easy to crack any code. (C) • [I am able to] recognize relationships between small whole numbers to solve a code. (E)

Continued on next page

Categories	Perspectives
Pattern finding	• even complex patterns can be understood in simpler ways. (F) • [I have learned that it is not true that I] am bad at finding pattern.s (B) • I learned that I can't find patterns really fast. (C)
Interest in CS	**Interest in CS** • Somewhat find this interesting. (G) • Actually find computer science and algorithms quite interesting. (H) • Find this topic interesting. (I) • [I have learned that it is not true that I] dislike data science. (G) • [I have learned that it is not true that I] dislike cryptography. (I)
Self-efficacy	**Self-efficacy** • I can be a hacker. (A) • [I] am able to collaborate and persist when facing challenges. (B) • [I] need to consolidate my basic math skills. (D) • [I] am able to develop methods and recognize relationships between small whole numbers to solve a code. (E) • [I can] solve/decrypt these kinds of messages quite easily, using skills I already have. (F) • [I have learned that it is not true that I] do not have enough pattern recognition to get into a career of computer engineering and science. (H) • [I] am able to do the tasks that we did during the lessons and I understood that I am able to do it. (J) • [I have learned that it is not true that] I'm not able to understand a brand new topic in 2 hours. (K) • [I have learned that it is not true that I] have a wholesome understanding of mathematics. (A) • [I have learned that it is not true] that I'm not able to understand the basics of cryptography. (B) • [I have learned that it is not true that I] am not able to understand cryptography. (J)
Collaboration	**Collaboration** • [I have learned that I] am able to collaborate (B)

Continued on next page

Categories	Perspectives
Collaboration	• Decrypting message mathematically is quite fun interesting to do with a group. (F) • I learned that even in math you can work together with other people. (J) • I have learned that teamwork is dreamwork. (K) • I can sometimes listen to the other's opinion instead of calculating everything myself. (K)

Chapter 6

Discussion and Conclusions

6.1 Computational Thinking Development

With the main research question of this thesis, we were looking to examine the extent to which teaching public key cryptography in high schools can serve as an enrichment to CT. In the learner reports, students have mentioned multiple times skills that are connected to CT.

There was a student who learned about finding ways to understand complex patterns in simpler ways, which suggests the discovery of using reduction to simplify problems. Wing (2006) named reduction as one of the strategies involved in solving a problem using CT.

A recurring theme in the data was pattern finding, as many students mentioned it either as the focus of the lesson or in regard to their own ability to find patterns. Pattern matching was mentioned by Wing (2010) as a distinctive skill that falls under CT.

When describing their learning, students mentioned just methods of finding solutions, but no specific results. In the learner reports, they mentioned strategies for finding the private key they discovered in the lesson or thought about their ability to develop methods for breaking protocols. That aligns with the definition of CT given by Aho (2012), which stressed that solutions are given in the form of algorithms. Similarly, (Nardelli, 2019) mentioned that one of the differences between mathematical and CT is in how the result is viewed. Therefore, as students have presented their learning as a method of finding solutions, they have engaged CT.

6.2 Learning Graph Theory through Cryptography

An additional question was to examine the extent to which students can gain knowledge about graph theory by understanding cryptographic graph-based protocols. In the data, there was only one mention of the tasks used for the transfer of knowledge to graphs. However, the student was thinking about applying logic to the fire hydrant problem, and there was no mention of a connection to the previous part of the lesson. It seems that the transfer of knowledge to graphs was not successful.

6.3 Other Findings on Teaching Cryptography in High School

From the data, we also saw that the activity had multiple beneficial effects for students as they became more digitally literate and were prompted to consider new careers.

Our results showed that the lesson led students to think about how well cryptography can protect their data. They learned what the purpose of cryptography is and discovered that it cannot provide a full guarantee of protecting their data. Similar to Lindmeier and Mühling (2020), we saw that it is enough to just give a basic overview of cryptography without explaining real-life protocols in detail to increase students' digital literacy. As students are now aware that there is always a way to crack the encryption, though usually it is time-consuming, they can now make better-informed decisions when sharing their data online. That was the result Percival et al. (2023) was looking for by recommending public key encryption to be taught already at middle schools, but we can see that at high school, it also had a positive effect.

From the data, we can also see that many students made discoveries about mathematics and saw real-life applications of mathematics during the lesson. We speculate that this will now give them more motivation to study it further, as they no longer see it as just an abstract theory with no use. Instead, they realised it can be used to protect their data, and with enough mathematical knowledge, they can understand how. Similar to Koblitz (1997), we have found that cryptography can be used for teaching new mathematical concepts, as students discovered new parts of mathematics and found new ways to use the knowledge they already had through the lesson.

After the lesson, students' interest in CS seems to have risen, as some students have discovered that they like it and started considering a career in CS. As Dagiene and Stupuriene (2016) said that students often see CS just as programming, we hypothesise that learning about cryptography might have shown them the broadness of CS. They saw that to become a computing scientist, they do not just have to code, but can also study many other areas, such as cryptography. Furthermore, computing scientists are often seen as the anti-social nerds, who sit behind a computer all day (Dagiene and Stupuriene, 2016), so the spiked interest in CS might also be coming from students realising that even in CS, they can collaborate with others. In the data, students mention surprise about being able to work with others in mathematics, but we speculate that they might have generalised that also for CS, as in the lesson, cryptography was introduced as falling more under CS than mathematics. Therefore, as the stereotypes about CS have been lifted, students might have felt inspired to consider it as a career.

One unexpected finding that was not mentioned in previous literature was how students' self-efficacy was influenced by the lesson. They gained confidence in their ability to tackle new topics. Moreover, students realised that the skills they already have are more useful than they expected when they could apply them to solve practical problems. They also realised the limitations of what they know so far and saw that there is still a lot more for them to learn in mathematics.

6.4 Limitations, Recommendations and Future Work

As the sample of this study only included 12 students, the results of this study are difficult to generalise and might be hard to reproduce. All students also came from the same mathematical class, so it is not possible to predict if the lesson would have the same influence if taught to students with different previous background knowledge.

Another limitation was that results for research were collected only via learner reports, and the lesson was not recorded. From the recordings, there might have been more insight into

students' thinking and how they developed CT. It would be especially useful to record classroom discussions that were part of the lesson, as they yielded interesting contributions from students. Moreover, the learner reports are subjective, as they are filled in by students, so it would be useful to examine how well they understood the contents of the lesson in an objective way.

Therefore, in a future study, the lesson should be replicated with more students from different schools to further examine the influence that teaching cryptography has on high school students. The lessons should be recorded to provide more information on student thinking and the process of understanding cryptography.

Furthermore, as Bell and Lodi (2019) have found that for best results, unplugged activities should be followed by plugged ones, so in this research, the full potential of the unplugged approach was not reached. Consequently, in future studies, more lessons should be dedicated to teaching cryptography so that students can also try out the cryptographic protocols on a computer to get an experience even closer to the real-life cryptography implementation.

From the results of this study, it seems that teaching cryptography in secondary education has many beneficial effects for students. When students learn about cryptography earlier in life, it makes them aware of how their data is protected. It also spreads knowledge about cryptography among more people, as otherwise, people would only know about it if they went on to study CS at university. Therefore, we would recommend more research into the ways of how to include teaching cryptography in secondary education and to further examine its influences on students.

Bibliography

Aho, A. (2012). Computation and computational thinking. *The Computer Journal*, 55:832–835.

Bandura, A. and Wessels, S. (1997). *Self-efficacy*, volume 10. Cambridge University Press Cambridge.

Bartzia, E.-I., Lodi, M., Sbaraglia, M., Modeste, S., Durand-Guerrier, V., Martini, S., al An, and Durand-guerrier, V. (2024). An unplugged didactical situation on cryptography between informatics and mathematics. *Informatics in Education*, 2023:25–56.

Bell, T., Alexander, J., Freeman, I., and Grimley, M. (2009). Computer science unplugged: school students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology*, 13.

Bell, T. and Lodi, M. (2019). Constructing computational thinking without using computers constructing computational thinking without using computers. *Constructivist Foundations*, 14:342–351.

Bell, T., Thimbleby, H., Fellows, M., Witten, I., and Koblitz, N. (1999). Explaining cryptographic systems to the general public.

Bell, T., Witten, I., and Fellows, M. (2015). Cs unplugged: An enrichment and extension programme for primary-aged students.

Bell, T., Witten, I. H., and Fellows, M. (1998). *Computer Science Unplugged: off-line activities and games for all ages*.

Calcagni, A., Lonati, V., Malchiodi, D., Monga, M., and Morpurgo, A. (2017). Promoting computational thinking skills: Would you use this Bebras task? In *Dagienė, V., Hellas, A. (eds) Informatics in Schools: Focus on Learning Programming*, pages 102–113. Springer International Publishing.

Dagiene, V. and Stupuriene, G. (2016). Informatics concepts and computational thinking in K-12 education: A lithuanian perspective. *Journal of Information Processing*, 24:732–739.

De Groot, A. (1980). Learner reports as a tool in the evaluation of psychotherapy. *Psychotherapy, research and training*, pages 177–182.

Diffie, W. and Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654.

Feaster, Y., Segars, L., Wahba, S., and Hallstrom, J. (2011). Teaching CS unplugged in the high school (with limited success). pages 248–252.

Fellows, M. and Koblitz, N. (1992). Kid krypto. In *Advances in Cryptology - CRYPT0 '92*, pages 371–389. Springer-Verlag.

Fellows, M. R. (1991). Computer science and mathematics in the elementary schools.

Fellows, M. R. and Koblitz, N. (1994). Combinatorially based cryptography for children (and adults). *Congressus Numerantium*, 99:9–41.

Grover, S. and Pea, R. (2013). Computational thinking in k–12 a review of the state of the field. *Educational Researcher*, 42:38–43.

Hermans, F. and Aivaloglou, E. (2017). To Scratch or not to Scratch? A controlled experiment comparing plugged first and unplugged first programming lessons. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education*, WiPSCE '17, page 49–56, New York, NY, USA. Association for Computing Machinery.

Joshi, A. and Joshi, B. (2011). A randomized approach for cryptography. In *2011 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC)*, pages 293–296.

Katai, Z., Toth, L., and Adorjani, A. K. (2014). Multi-sensory informatics education. *Informatics in Education*, 13:225–240.

Keller, L., Komm, D., Serafini, G., Sprock, A., and Steffen, B. (2010). Teaching public-key cryptography in school. In Hromkovič, J., Královič, R., and Vahrenhold, J., editors, *Teaching Fundamentals Concepts of Informatics*, pages 112–123, Berlin, Heidelberg. Springer Berlin Heidelberg.

Koblitz, N. (1997). Cryptography as a teaching tool. *Cryptologia*, 21:317–326.

Kooloos, C. C. (2022). *Eye on variety: The teacher's work in developing mathematical whole-class discussions*. [PhD thesis, Radboud Universiteit Nijmegen].

Lindmeier, A. and Mühling, A. (2020). Keeping secrets: K-12 students' understanding of cryptography. In *ACM International Conference Proceeding Series*. Association for Computing Machinery.

Nardelli, E. (2019). Do we really need computational thinking? *Commun. ACM*, 62(2):32–35.

Percival, N., Narain, S., and Lee, C. S. (2023). Modern cryptography education of middle school students: A review of current works. In *SIGITE 2023 - Proceedings of the 24th Annual Conference on Information Technology Education*, pages 33–38. Association for Computing Machinery, Inc.

Rosamond, F. (2012). *Passion Plays: Melodramas about Mathematics*, pages 80–87. Springer Berlin Heidelberg, Berlin, Heidelberg.

Rosamond, F. (2018). Computational thinking enrichment: Public-key cryptography. *Informatics in Education*, 17:93–103.

Súst rik, M. (2013). Public key encryption for kids. [Blog post].

Taub, R., Armoni, M., and Ben-Ari, M. (2012). CS unplugged and middle-school students' views, attitudes, and intentions regarding CS. *ACM Trans. Comput. Educ.*, 12(2).

Thies, R. and Vahrenhold, J. (2013). On plugging "unplugged" into CS classes. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE '13, page 365–370, New York, NY, USA. Association for Computing Machinery.

Van Kesteren, B. (1993). Applications of De Groot's "learner report": A tool to identify educational objectives and learning experiences. *Studies in Educational Evaluation*, 19(1):65–86.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49.

Wing, J. M. (2010). Computational thinking: What and why?