DEGREE PROJECT IN TECHNOLOGY,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, SWEDEN 2019

Implementing a Nudge to Prevent Email Phishing

Viktor Vitek Taqui Syed Shah

KTH ROYAL INSTITUTE OF TECHNOLOGY
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Abstract

Phishing is a reoccurring issue, which uses social engineering as an attack strategy. The prevention of these attacks is often content-based filters. These solutions are however not always perfect, and phishing emails can still be able to get through the filters. We suggest a new strategy to combat phishing. The strategy is a technical platform which uses the psychology concept *nudge*. Nudge is a concept that can be used to change a certain behaviour, in this case to make people more cautious when reading their emails.

The objective of this thesis is to suggest a nudge using a technical platform regarding possible desensitization. The nudge aims to change email related behaviours to more healthy ones. To get indications if the nudge has benefits, a qualitative survey was made. When using a psychology-based solution, one must address the possibility of desensitization. To minimize possible desensitization, a quantitative analysis was made where different ways to minimize desensitization were assessed. Data for this analysis was gathered by a simulation modeling, where the simulation aimed to replicate a user performing email related events.

The conclusion of the simulation results showed that a whitelist approach was the most appropriate for our nudge. The approach minimized the chance of possible desensitization while having a low risk of not performing a nudge when needed. The conclusion of the survey results was that there was an indication of behavioural change and that there existed a risk of possible desensitization.

Keywords

Hacking; Phishing; Social Engineering; Psychology; Nudge;

Sammanfattning

Nätfiske är ett återkommande problem, som använder sig av social manipulation som attackstrategi. Försvar mot dessa attacker är ofta innehållsbaserade filter. Dessa lösningar är inte alltid perfekta, då nätfiske kan ibland gå förbi filterna. Vi föreslår en ny strategi för att bekämpa nätfiske. Strategin är en teknisk plattform som använder det psykologiska konceptet *nudge*. Nudge är ett koncept som kan användas för att ändra ett visst beteende, i detta fall för att göra människor mer försiktiga när de läser sina emails.

Syftet med detta arbete är att föreslå en nudge i en teknisk plattform där man tar hänsyn till eventuell desensibilisering. Nudgens mål är att ändra email-relaterade beteenden så att beteendena blir säkrare. En kvalitativ undersökning gjordes för att få indikationer om nudgen har möjliga fördelar. När man använder en psykologibaserad lösning så måste man ta itu med möjligheten av desensibilisering. En kvantitativ analys gjordes där olika sätt att minimera desensibilisering bedömdes, för att sedan kunna minimera desensibiliseringen. Data för denna analys samlades in genom en simuleringsmodellering, där simuleringens syfte var att replikera en användare som utför email-relaterade händelser.

Slutsatsen av simuleringsresultaten visade att en whitelist-metod var den mest lämpliga för vår nudge. Metoden minimerade risken för möjlig desensibilisering, samtidigt som den hade en låg risk att inte utföra en nudge när det behövdes. Slutsatsen av undersökningsresultatet från enkäten var att det fanns en indikation för beteendeförändringar och att det fanns en risk för eventuell desensibilisering av nudgen.

Nyckelord

Dataintrång; Nätfiske; Social Manipulation; Psykologi; Nudge;

Acknowledgements

We want to thank our examiner Peter Sjödin and our supervisor Markus Hidell. They helped us tremendously throughout our project and always kept us in the right direction. Thanks to Christian Günther-Hanssens from Cogito Credo for the idea behind the project as well as recommending literature. We want to thank all participants who helped in the survey. A grateful thanks to Nuo Chen and Lovisa Josefsson for being opponents and providing good feedback.

Authors

Viktor Vitek vitek@kth.se
Taqui Syed Shah taqui@kth.se
Information and Communication Technology
KTH Royal Institute of Technology

Place for Project

Stockholm, Sweden
KTH Royal Institute of Technology, Campus Kista, Kistagången 16

Examiner

Peter Sjödin
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology

Supervisor

Markus Hidell
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology

Contents

1 Introduction

We humans have become more dependent on the Internet than ever before. This creates possible vulnerabilities where the security might be at risk. One attack which is abusing the dependency of the Internet is the attack called phishing, which is a form of social engineering that manipulates users to give up their credentials. Credentials refers to bank account numbers, social security, location or password for services such as Google. Current solutions solve this issue by using detection tactics that use Bayesian filtering, but hackers are becoming cleverer with their methods and therefore can detection tactics become obsolete very fast. We suggest a new prevention method against phishing. This form of prevention uses psychology to make humans more aware of phishing emails. A nudge can help people become more aware and the report will focus on how to implement a nudge to prevent phishing. One needs to consider the possibility of desensitization when working with psychology-based solutions, which is defined as the reduced sensitivity of a certain stimulus, in this case the effect of the nudge.

1.1 Background

Today, there are many victims that have fallen for the hacking method phishing. An example of a phishing attack that has received much attention is the one that happened during Hillary Clinton's campaign in 2016, where the campaign chairman John Podesta was hacked because of a phishing email [29]. A Danish information security report from 2018 states that a bank in Denmark has lost around 5.5 million DKK because of successful phishing attempts. This was more than the losses due to digital burglaries [27].

Phishing attacks are a problem for both large companies and small companies [35]. A solution to prevent the attacks could be to draw the attention of the email users towards phishing email traits. One could implement the solution by developing a software that nudges the users when something suspicious is around the corner.

1.2 Problem

The research question for this thesis is the following;

"How to implement a nudge against phishing with regard to possible desensitization?"

Before our implementation, a nudge against phishing attacks has not been implemented in a technical platform. Under the development of the implementation, one should consider the possibility of desensitization, as it is a psychology-based solution.

The following bullet points present the problems that one must address before answering the research question.

- Identify and define the characteristics of phishing.
- Identify the principles of a nudge.
- Implement a possible solution for phishing prevention using a nudge.
- Evaluate the implementation of the chosen solution.
- Implement and evaluate strategies for minimizing the amount of desensitization that occurs.

1.3 Purpose

The purpose of the report is to identify the common traits of phishing and then find a way to nudge people into being more aware of phishing while minimizing the possibility of desensitization. The purpose of the work is to lower the amount of successful phishing attempts.

1.4 Goal

The goal of the work is to create a software that lowers the number of successful phishing attacks. The software should use a nudge to combat phishing and reduce the possibility of desensitization. The implementation will create awareness and

nudge users into more cautious behaviours when viewing emails. Another goal is to create a phishing software that will be studied by the Danish authority "Erhvervsstyrelsen". Erhvervsstyrelsen is the Danish business authority, which helps businesses to run in Denmark [11]. The authority will test the software in a controlled environment with the goal to determine if the email viewing behaviour changes or not. See more about the test in the Future Work, Chapter 8.

1.5 Benefits, Ethics and Sustainability

Ethical issues that are worth noting are those that one needs to consider when working with psychological concepts. One could argue that one restricts the free will with a nudge. The intention of the nudge matters, in this case, the nudge should be made with good intentions, since the aim is to minimize phishing attacks. Hopefully, this will work towards a more socially sustainable world.

The software should not do any surveillance or have any possible security flaws, as in the wrong hands the software could have access to the emails of the users.

With regard to the social sustainability aspect of human rights, respect for the survey participants was held. The autonomy, decision-making and dignity of participants were all respected during the survey.

1.6 Methodology

The research approach will be in a deductive manner as we formulate our own hypothesis on an existing theory and then design a research strategy to test it. The test result will be examined and compared with our findings from the literature [12]. The research method will be a qualitative research, as it concerns understanding the meanings and behaviors to reach a tentative hypothesis and develop computer software [20]. The purpose is defined by applied research as we aim at finding solutions for our hypothesis, but more of a conclusive research than an exploratory as we test our hypothesis.

Our research process will contain a general narrative literature review. The review summarizes the most important aspects of the current state-of-the-art knowledge

of the concepts nudge and phishing. From the review one can draw conclusions and identify inconsistencies [13]. The summarized information from the review can be used as an introduction to the thesis.

1.7 Stakeholders

The idea behind this thesis comes from a startup company called Cognito Credo. Cognito Credo aims to develop a cyber security concept for email using the principles in psychology. They had an idea that they won a contest with. The contest was called ”Cybersecurity Challenge” [10] and was made by the Danish authority Erhvervsstyrelsen [11]. One can read about the idea in depth in on their website [19].

1.8 Delimitations

The scope of this project will be limited to phishing attempts on browser-based email clients. The reason for this limitation is that the more dangerous phishing attacks target large corporations. Another delimiter is user independent solutions to prevent phishing. As mentioned, there exists many solutions to phishing. The new suggested solution is a user dependent one and the project will focus on those.

About nudges, the study does not study ways that forcefully change behaviours. Examples of forceful ways are laws and bans. In our case, it would be to restrict the user to view or open dangerous emails. A nudge must present the freedom of choice. More about the definition is presented in the Chapter Theoretical Background 2, Section 2.6.

1.9 Outline

The following list describes the overall structure for the thesis.

- Chapter 2 covers the state-of-the-art knowledge of the concepts nudge and phishing. The information here is the results from the literature review.

- Chapter 3 has a description of the engineering-related contents that was used to solve the problem, such as the methodologies and methods.
- Chapter 4 has a description of all the design choices for the implemented nudge. The implementation of a survey and a simulation is also described.
- Chapter 5 shows the results from the survey and the simulation.
- Chapter 6 gives the discussion about the results. The strength and weakness of the nudge are discussed, as well as the possible benefits and desensitization.
- Chapter 7 shows a conclusion drawn from the discussion which answers the thesis research question.
- Chapter 8 consists of a future work where one will find possible improvements to the work and what the next step is for the plugin that was created.

2 Theoretical Background

This chapter contains the information necessary to understand the thesis. It focuses on defining the concepts nudge, phishing and the design choices made for the software.

There are many types of phishing attacks and there exist multiple ways of preventing them. Nudge is a newly coined term and there are different definitions. It is important to gather information from several studies to give an accurate representation of what phishing and a nudge entail. The goal is to find the optimal nudge against phishing, which should trigger at the appropriate times.

2.1 Security Hacking

Security hacking is exploiting vulnerabilities in a computer system or a network [3]. Security hacking attacks can be classified into two different branches, user-dependent attacks and user-independent attacks. User-dependent attacks requires an action from the user, whereas user-independent attacks do not require an action from the user. An action can be clicking a malicious link or installing a malicious software [3]. In the user-dependent attacks the vulnerability that is exploited is targeting the vulnerabilities of the human psychology. Humans are often lured into performing certain actions in user-dependent attacks. Drawing users attention towards possible hacking attempts can reduce successful security hacking attempts.

2.2 Phishing

Phishing is a form of user-dependent attacks. In phishing, victims are contacted and are lured into giving information such as personal information or banking information. Phishing emails often consist of either a malicious link or attachment [37]. Phishers can be able to gain information about the user or able to gain control of a user's machine. Phishers can convince users to perform certain actions by using social engineering. Social engineering is when one exploits the

human's trust to gain personal information and using it for malicious purposes [3]. Social engineering is used for two purposes in phishing. Firstly, to create credibility in the phishing email. Secondly, to create a feeling of urgency [37]. Credibility enables users to trust the content of the email and urgency tricks users into completing an action quickly [6].

The thesis will focus on two different types of phishing attacks. The first one is deceptive phishing and the second one is spear phishing. Deceptive phishing is when phishing emails imitate reputable companies. Phishers use the credibility of a reputable company to steal user's personal information, credit card information or login credentials. The success of deceptive phishing is dependent upon how much the phishing email resembles a legitimate email from a reputable company [6].

Spear phishing is an email attack that includes personal information such as a name or an email address of someone known to be friendly to the victim, to ensure credibility [6]. Spear phishing often targets certain people within a company and can therefore be more deceptive than a regular deceptive phishing email.

2.3 Existing Phishing Prevention

Phishing prevention is often done by detecting certain characteristics of phishing emails [42]. The first characteristic is an HTML-based email [16]. HTML stands for Hypertext Markup Language [16] and is the building block for modern websites. HTML elements describe the structure of the website, which includes the text placement, font, text size, picture placement and more. HTML elements are described by tags, which are not displayed in the browser. Instead, browsers use the tags to render the content of the page.

Phishers hides information from the user by using the tag called anchor. Anchors are used to redirect a user to a web page without explicitly showing the link [16]. An example would be when a link would open a fake PayPal website, instead of the real PayPal website.

The second characteristic is imitating a reputable company. This is done to

create credibility in the email and trick users into trusting the content of the email. The third characteristic is asking for specific information, such as personal information. By studying these characteristics, one can create programs that looks for these and blocks them.

2.4 Email Spoofing

Email spoofing is when one is creating an email with a forged sender address [26]. Phisher uses email spoofing to impersonate someone within a company to ensure credibility and then trick people into performing a certain action. For example, tricking them to giving up their credentials or installing malicious software. There are two methods of email spoofing. One is email address spoofing and the other one is the display name spoofing [40]. Email address spoofing is when the phishers are spoofing the actual email address. Display name spoofing is when the phishers are spoofing the name that is displayed in the email client.

2.5 Spam Filters

Spam filtering is a process of organizing emails based on a specified criterion, if they are recognized as spam or not. Understanding the techniques that are used in spam filtering can help one when implementing a detector for phishing emails. There exist two main categories within email spam filtering, which are content-based filters and list-based filters. Each category has a different method of filtering emails.

2.5.1 List-Based Filters

List-based filters are used to minimize spam emails by categorization. These categories are dependent on the email address.

Blacklists are pre-set lists of email addresses. Emails from these email addresses are considered as spam [33]. Received emails are categorized into spam and non-spam by comparing the email address of the sender to the email addresses

in the blacklist. Adding a new email address to a blacklist is called blacklisting. Spammers with new email addresses needs to be manually added to the blacklist. A blacklist can be controlled by a third party, but then it is labeled a real-time blackhole [33].

Whitelists are also pre-set lists of email addresses [33], but mails from these email addresses should be safe. Like blacklist, a whitelist categorizes received emails into spam and non-spam. However, a received email is considered as spam if the email address is not in the whitelist [33]. Adding a new email address to a whitelist is called whitelisting. The whitelisting process can also be done by a third-party, like the real-time blackhole lists [33].

2.5.2 Content-Based Filters

Like list-based filters, content-based filters are used to minimize spam emails by categorization. The content-based filter's categorization is based on the words used in the email and the overall structure of it.

Word-based filters are content-based filters which categorize emails containing certain terms as spam. A wordlist is a list containing all the terms that can give an indication of a spam email. Spammers can get past these filters by misspelling words to get through word-based filters. Therefore, to combat the spammers, the wordlist must be updated with time, by increasing the list with more spam-related words.

Heuristic filters are also content-based filters, that use various algorithms to analyze the content of emails [33]. These algorithms assign a score to the email, based on the content of an email. If the email is scored higher than a certain threshold, it is considered as spam. For example, if an email has the words "free", "viagra" or "rolex" in it, then the heuristic filter will give the email a high valued score. Heuristic filters work fast and can be effective at minimizing spam. However, experienced spammers try to avoid including certain words, to bypass the filter by getting a lower score from the heuristic filter.

Bayesian filters are content-based filters that employ statistics and probability for determining spam [33]. To effectively block spam, the Bayesian filters must

initially be trained by a person who is manually marking each message as either spam or legitimate. Bayesian filters make a probabilistic suggestion, for example, if a “vicodin” occurs 67 times in spam emails, but only two times in legitimate emails, the filter will assume that emails containing the term ”vicodin” are spam.

2.5.3 Spam Filtering Methods in Phishing Prevention

Spam emails can be of many types, but commercial and phishing emails are the most common ones [42]. The commercial type is the most common one, and because of that, most spam filters focus on preventing the commercial spam, which makes it easier for phishing emails to get past the spam filters. [42].

Spam filters do not always detect phishing because of the differences in wording choices between phishing emails and commercial emails. However, the techniques used in spam filtering can be used and applied in detecting phishing emails. There are four techniques used in spam filters that can also be used in the phishing detection, which are blacklisting, whitelisting, heuristics and Bayesian filters [42].

Heuristics and Bayesian filters do not always guarantee results. These filters are content-based filters, therefore, phishers can pass these filters by changing the content of an email. Content-based filter needs to be continuously updated since phishers can change the email content [17].

2.6 Nudge

In this section the concept of nudge will be discussed, which includes relevant background information to the thesis.

2.6.1 About the Nudge

A nudge is a behavioural psychological method that by soft-paternalistically showing tries to change to a behaviour of a specific target [32]. Meaning that in a

non-forceful way the nudge hints at a choice one wants the target to make, without them realizing. These nudges can be political, lucrative or environmental. The intentions behind them are often good, but there exists cases where the intentions are ”bad”, for example when they are used in gambling. The concept of nudge was developed by two scientist, Cass R. Sunstein and Richard H. Thaler [41]. Sunstein and Thaler's definition of a nudge is the following: *“A nudge, as we will use the term, is any aspect of the choice architecture that alters people's behaviour in a predictable way without forbidding any options or significantly changing their economic incentives. To count a same renudge, the intervention must be easy and cheap to avoid. Nudges are not mandates. Putting fruit at eye level counts as a nudge. Banning junk food does not.”* [41].

There are many types of nudges, but nudges never restrict the freedom of choice. That means that a nudge does not forbid, nor temper with economic incentives. The definition of a nudge made by Sunstein and Thaler is the most commonly used definition. For a more detailed definition, one could use the one made by Hansen [21]. He agrees on multiple points, but a difference from the Thaler and Sunstein definition is that Hansen points out that the nudge is an action made by a creator, not something that occurred by itself. A nudge may not successfully change behaviours, but it would still count as a nudge, but a failed one. The important thing is that the nudge is created by someone and not something that occurred naturally for it to be defined as a nudge.

The Table 2.1 on the next page is showing different ways to change behaviours.

Type	Description	Source
Nudge	Changes the behaviour of people in a way without forbidding any options or significantly changing the economic incentives. Described in detail above.	Thaler and Sunstein [41]
Laws or Bans	Making something illegal may force a behaviour change, but it is an obvious and forceful way to change a behaviour and it is not considered to be a nudge by the agreed definition.	Thaler and Sunstein [41], Johnson et al., 2012 [25]
Economic measures	For example, taxes, subventions or lowering prices. Not considered as a nudge with the same motivation found in ”Laws or Bans” above.	Thaler and Sunstein [41], Johnson et al., 2012 [25]

Table 2.1: Table about behaviour changes

Kleef and Trijp mentions that the concept nudge generally assumes that the decision of choice is based on how people perceive, think, and decide. They also state that the decision is based on automatic and learned responses, intuition, habits and unconscious associations. A good nudge would try to target these factors of the decision-making to try to change a behaviour. A nudge would try to change the decision so the affected would pick the ”right” choice, where the definition of ”right” is set by the creator of the nudge. Kleef and Trijp states that a nudge is extra useful when it is a complicated choice that needs to be made, where the choice has long term benefits which could be hard to spot, or when feedback from the choice is lacking. [14]

There are many types of nudges and they often affect people without them knowing. Nudges fall into two different categories; Priming or Salience nudges, or a combination of both. According to Blumenthal-Barby & Burroughs [5], the priming nudges influence people by altering aspects of the environment, while the salience nudges provide the relevant information to the affected to make them

decide on a certain choice. The Table 2.2 on the next page shows different types of nudges.

2.6.2 The science behind the Nudge

Cognitive psychology is the science of how the mind produces and realizes intelligent thoughts. Human thought mechanisms play a major role when one wants to understand the reasons for different types of behaviours. A nudge is a part of the study behaviour economics, where one studies to understand how decisions are made. Cognitive psychology is the foundation for all the social sciences and has a lot of practical implications in our daily lives. The study on the human condition has only been actively studied for the last 150 years. The cognitive psychology overthrew behaviourism as it could offer three things that behaviourism could not. Cognitive psychology could help with practical issues, get computers to behave intelligently (e.g. AI) and thirdly, the linguistic studies. [1]. A memory that is often practiced is strengthened, accordingly to Anderson as he writes in his book *Cognitive Psychology and Its Implications* [1]. Something to keep in mind when someone is implementing a nudge.

Type	Description	Source
Default	By setting a choice as the default one, it will automatically be the easiest choice to make as one does not need to consider the other options. According to Blumenthal-Barby & Burroughs, consumers tend to choose default options. This would be an example of a priming nudge.	Blumenthal-Barby & Burroughs [5], Lehner et al. [28]
Framing	A framing nudge is a priming nudge where the choice options are framed into making one choice more attractive and therefore more likely to be chosen.	O'Brien & Davies [30], Lehner et al. [28]
Norms	By telling people that the majority is picking a certain choice, they may feel the urge to "fit in" and choose the same choice. An example of this is how the British government made people pay their taxes on time. They sent out letters that said, "most of your neighbours pay their taxes on time". This led to an increase in the number of people that paid on time [36]. This is an example of a salience nudge.	Owain Service et al., 2014 [36], Lehner et al. [28], Blumenthal-Barby & Burroughs [5]
Inform	A subtle, informative nudge can change a behaviour. An example is an environmentally friendly mark on a product. The affected may start to think about the positive effects and consider the marked product as a better option.	Thaler & Sunstein [41], Johnson et al., 2012 [25]
Changing area	A priming nudge is for example when one would place fruits on an eye level to grab attention. Another example would be to decrease the size of the plates to make people take less food per plate. The plate example successfully decreased the food waste. [7]	Reynolds et al., 2019 [7]
Decoy effect	An approach which is used in advertising is when one affects a choice of two different option, by adding a third one. The third one influences the choice of the other two. [24]	Huber et al., 1982 [24], Huei-Chen Hsu & Wen-Liang Liu, 2011 [22]

Table 2.2: Table about nudges

2.6.3 Desensitization

Desensitization is defined as the reduced sensitivity of a certain stimulus [9]. Desensitization could cause nudges to become less effective over time. When implementing a psychology-based solution this factor needs to be considered. The user needs to take their time to consider their options, but if the nudge becomes too repetitive it might desensitize the user. The desensitization would lead to a reduction of the desired impact and the nudge could become futile. The best way to reduce desensitization is to lower the number of nudges that happens in unnecessary times, which means that the nudge should not occur when there is not a real threat.

2.6.4 Regards to creating our Nudge

Nudges can often be implemented in various ways, as the Table 2.2 shows. The informative nudge can raise awareness about a choice to let the users act based on the information, which can be used against phishing. An informative nudge could raise awareness of the sender or the destination of a link. One can also present the informative nudge in many ways, for example with a hovering pop-up above an email address or a link. Other examples can be color coding the emails, links or addresses based on possible phishing attempts. One more example can be to initiate a dialog when a link is clicked instead of following through to the destination of the link. The dialog can display information about the sender or the link.

Stein and Bransford showed in their study [38], if one is constructing precise elaborations it may help to improve the memory, which should be considered when writing sentences for a nudge [38]. If one writes “This link may be harmful”, then one could add the elaboration “for your computer.” to help the users to remember the material.

Another concept which Anderson [1] brings up is the following: *“People are only*

able to show high levels of logical reasoning with modus ponens" ([1], page 277). Modus ponens is Latin and means a method for affirming. Its counterpart would be modus tollens which is a method for denying. Also, when it comes to decision-making people do not always make the choices that has the best expected value. They often value higher and safer chances of any rewards, more than wanting to gamble for the all or nothing, even if it has a higher expected value ([1], page 312).

Framing effects are shortly described in the Table 2.2. Anderson [1] explains that framing effects are when choices can be displayed in different manners, which can change the outcome of the choice dramatically. When asking a question, it can be framed in such a way that people tend to pick one alternative more, which one could use in a nudge.

In an interview with Cass Sunstein, the following question was asked: *Can nudges lead to lasting change?* He answered: *"There is good research on the circumstances under which using social norms result in persistent instead of short-term behavioural change. With respect to energy use, so long as people are frequently reminded, it works."* ([18], page 35). When creating a nudge, it would seem important to nudge frequently or by using the social norm nudge, which is described in the Table 2.2.

2.7 Related Work

This section describes studies that are related to the thesis. Every study listed here has a nudge or a psychology-based solution connected with IT. By reviewing the related work, one might avoid mistakes or learn from their achievements.

2.7.1 Nudge used in Twitter

FeedReflect is an example of an IT-related nudge, which is a tool that nudges people to question the credibility of twitter posts [4]. Their study suggested that their tool helped people assess the credibility of the news. The study followed 16 participants that were divided into smaller groups and then the software was

tested on them for three weeks. After the three weeks it was shown that students which had used the extension where more aware of possible fake news that occurred. If one could implement a nudge that could make people more aware of news credibility, that would indicate that it is possible to make a nudge that makes users aware and reflect over emails' credibility when using an email client.

2.7.2 Using nudges to improve online security behaviours

An interesting study [2] which addressed the main issues of IT consumerization, used nudges to improve social security, evaluated different forms of nudges with the purpose of increasing cyber security. They used different messages to see what promoted the best behaviour. The first was a coping message, which told them how to stay safe. The other one was a threat message, which told them the possible threats that might occur when doing an activity. The study showed that both nudges were effective at changing behaviours, but that the coping form lead to a more safe behaviour. This shows that using a nudge within cyber security is possible and the possibility of it being applied in phishing is therefore likely.

2.8 Research Method

As mentioned in the introduction 1.6, the research process contained a narrative literature review. From the review, conclusions were drawn with regard to the topic and it was helpful when identify gaps or inconsistencies in the state-of-the-art knowledge of the concepts nudge and phishing [13].

Data was collected from the literature review by reading sources from scientific reports and journals. Databases that were used were those that KTH has access to, which was Scopus, Xplore Digital Library, ACM digital library and IEEE Xplore. Google Scholar was also used for some basic information in the beginning to get a quick overview of the topic. Boolean formulas were used when searching to narrow the results. First, a search for ”nudge” and ”phishing” were done, both to get some scientific reports. The search expanded to “nudg* AND phishing”, “email* AND phishing”, “nudg* AND phishing AND behaviour”. An iterative

search was used in the sources. Meaning that if something was found interesting and relevant, its sources were used, to find the primitive sources. The database ScienceDirect was also used as it covers a lot of social sciences, which was useful when review the term nudge.

3 Methodologies and Methods

To be able to answer the research question one was first required to evaluate the possible behaviour changes due to the nudge and the desensitization effects. Secondly, one needed to evaluate the desensitization impact. Data was collected to be able to evaluate these.

Two different data collection methods were used for the respective evaluation segments. For the desensitization impact segment, the data was collected from a simulated test environment. A simulation was created where different strategies were trying to trigger the nudge at appropriate times were tested. A quantitative method was applied as the focus was on the percentage of interesting events for each different strategy. The purpose of this simulation test was to examine how often the nudge does trigger in a real setting. This was done to later be able to answer the questions about annoyance and possible desensitization. More about the implementation of the simulation in the Section 3.3.

A survey was made to evaluate the possible behavioural changes due to the nudge. A survey is classed as a qualitative method. Survey Monkey was used to create the survey, as it is easy to use, and the questions can be asked in many different formats. More about the survey in the Section 3.4.

When analyzing the data received from the tests and the survey, the goal was to turn the data into meaningful information. Microsoft's Excel program was used to create the more advanced graphs and tables which seemed most appropriate according to the research objective. With the graphs and tables one can find the possible trends easier.

3.1 Development Method

The models of the software program were made from the conclusions of the literature study. Test driven development was used during the development. Requirements and test cases were created. The test cases verified that the program meets the requirements of the model.

The development method was the agile method called SCRUM [34], but modified to make it simpler. No roles were assigned as one commonly do when working with SCRUM, as the group only contained the authors.

3.2 Implementation

In this section both the software structure and the nudge implementation's methods and models will be shown.

3.2.1 Software

At the beginning of the project, a basic schematic was created for the overall functionality of the program. The program was described with a JSP-diagram [23]. On the next page, the picture of the diagram is shown, which is the Figure 3.1. The diagram is built with three different parts, which were sequence, selection and iteration. Sequence is represented as a square, a selection is represented as a square with a smaller circle in the right corner and an iteration is represented as a square with the star symbol in the right corner. When reading a JSP-diagram one reads the top level of the diagram first, from left to right. Then from top to bottom for more details about the program.

The software starts with identifying an email. In which the filtering strategy is applied, which is described more in the Section 3.2.2. If the email address is considered a possible threat, then a link listener is added to each link in that email. The last part is the nudge part, which activates when a link is clicked. The dialog asks the user who they think the email is from. A scrambler generates similar email addresses to the original sender. Then the user needs to answer which email address they think is the correct one. Read more about how the scrambler works in the Section 3.2.4, and more in depth of how the nudge works in the Section 3.2.3. If the user answers correctly, the nudge will declare the email safe. If they answer incorrectly the nudge will advise the user to be cautious.

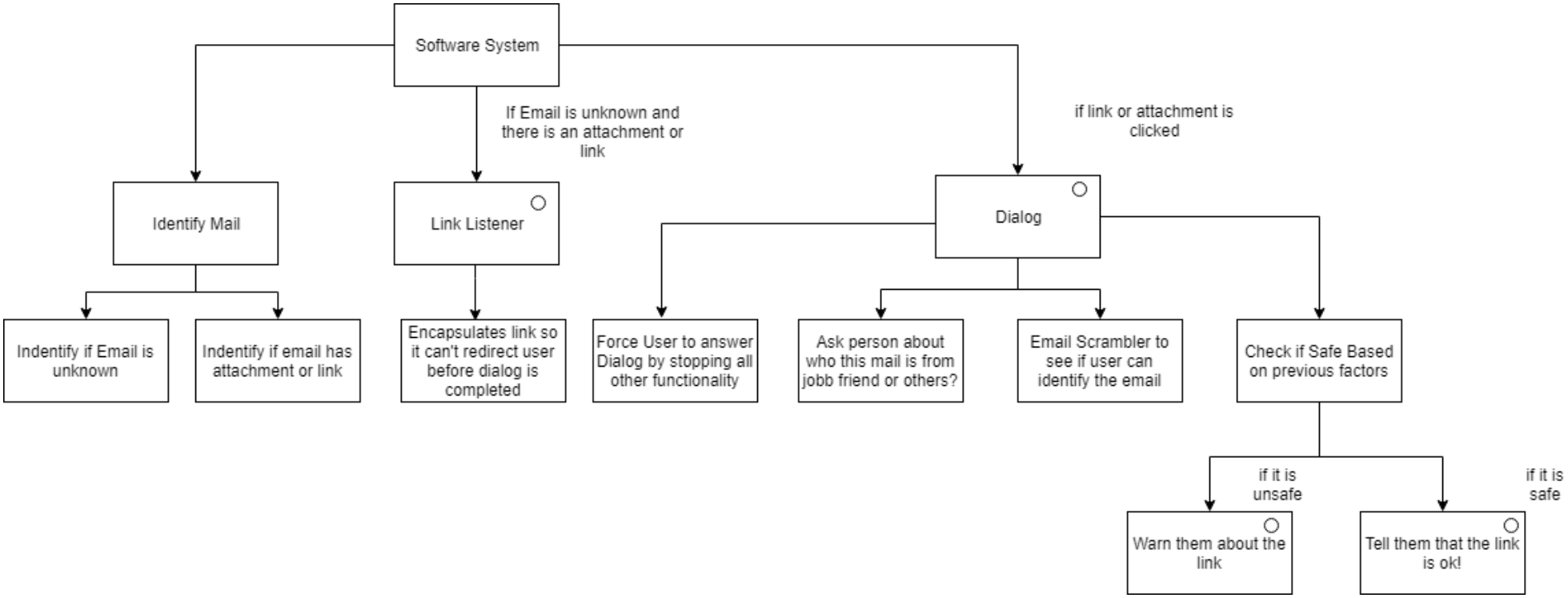

Figure 3.1: JSP Diagram explaining the overall structure of the software

3.2.2 Filters

Creating content-based detection is difficult. The reason is that if a phisher changes the content, then content-based filters needs to be updated. Email address-based solutions are an alternative. The email address-based solution is the basis for the nudge. To lower the possibility of desensitization, one needs to lower the frequency of non-useful nudge triggers. This can be done with list-based filters strategies.

A specific list-based filter was not implemented directly. Different types of filters were tested and evaluated in a simulated environment. The simulated environment is described in detail in the Section 3.3. The list-based filters that were tested were based on different whitelisting strategies.

- Whitelist
  The whitelist approach lets the users whitelist email addresses that the user considers to be friendly. The software triggers on all attachments and links that are clicked, until the user whitelists an email address. When an email address is whitelisted, the user can click on links and attachments on the email from the email address without triggering any nudges.
- Sentlist
  Sentlist is like the whitelist, but instead of letting the users choose which email addresses to whitelist, the software whitelist all the email addresses that the user has sent an email to previously.
- Double message strategy
  The double message strategy is that emails get whitelisted after triggering a nudge. The double message strategy will be referred to as the double strategy in the report.
- Whitelist + Sentlist
  This is a multi-strategy approach. This approach combines whitelist and sentlist, where the positives from both hopefully reduces the chance of desensitization.

3.2.3 Nudge

Based on the information gathered in the Section 2.6 of the theoretical background, the decision on how the nudge should be implemented was chosen. As said in the Section ”Regards to creating our Nudge” 2.6.4, there are many types of nudges, but the decision was to create an informative nudge that aims to raise awareness about the sender’s email address. This seems natural as it works good with the filters discussed in previous the Section 3.2.2. The presentation of the nudge was decided to be a dialog where the user must answer question about the sender. When a user clicks a link in an email the dialog would trigger. A dialog can be more aggressive choice compared to a color-coded presentation. Regarding the nudge, an elaboration of some sort is wanted to help the users to improve their memory of the nudge’s purpose.

The idea was that the dialog should contain two pages, where the first one should be the introduction page that presents and describes the nudge. The aim of the first page was to give a helpful description without presenting too much information, as this page will occur many times and might cause annoyance. Instead of only telling the user that they may be in danger, the focus was on them to verify that the sender really is the person or company who they think it is. The model of the dialog should have the heading ”Who is this from?” with a subheading ”Links from unknown users can be harmful, which may target your credentials and computer”. The elaboration in the subheading can help the users to remember the purpose of the nudge. The first page of the dialog’s model had checkboxes with categories of common types of senders, where the user can mark who they think the email is from. If they select the alternative ”This is spam/phishing.”, the dialog closes and will not proceed to the next page. But if they select any other alternative the dialog proceeds to page number two. See the Figure 3.2 on the next page to see the model of the dialog page number one.

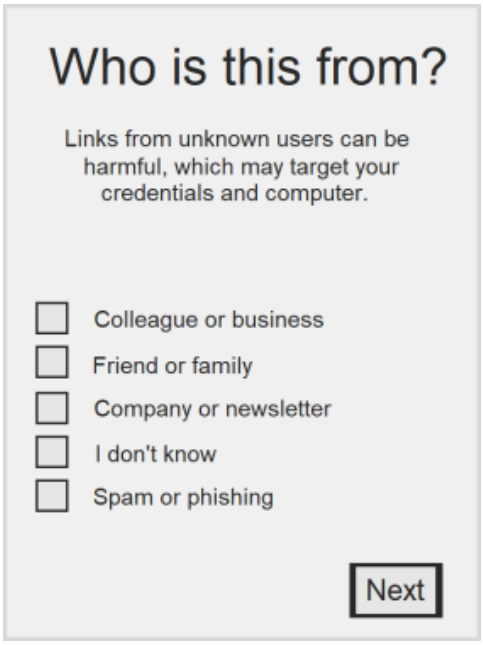

Figure 3.2: Dialog page 1, it contains a short description of what may happen and a form where the user enters who they think it is from.

The idea for the second dialog page was that it should contain a test, that would ask the user which address they think have sent the email, from a set of email addresses. This is one way to combat common phishing types, as the user might choose the email address which looks the most legitimate one, but if they pick the wrong one it may help the user to realize that it might be a phishing email. The focus of this page is to make the user aware of the sender. See the model of the second page in Figure 3.3 on the next page.

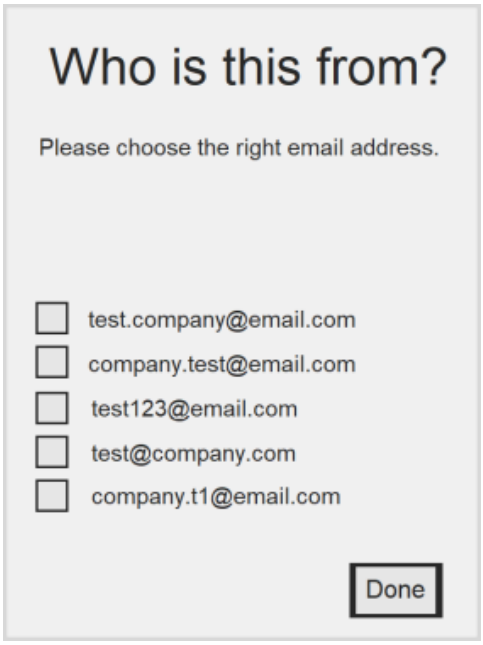

Figure 3.3: Dialog page 2, same heading as page 1, but with a new description which encourages the user to make a choice of which address they think is the real address of the sender.

The set of alternatives of the emails on the second page was generated to look alike the original received one. A model of the scrambler was built, that would create four new different addresses from the received one. More about the scrambler in the next Section 3.2.4.

3.2.4 Scrambler

The idea of the scrambler was that it should divide an email address into as many reasonable long parts as the address could be divided into. Then by combining the different parts one could generate multiple new fake addresses. An email address has a local part, an ”@” symbol and a domain. Different ways of combining these were limited as the ”@” symbol was needed to be placed in the middle. If the domain or local part contained a ”.” symbol, things could be scrambled up even more. Another way was to add numbers to the end of the local-part. These ideas created the requirements of the scrambler. All of these functions were implemented according to the requirements and related test cases to the scrambler.

3.2.5 Requirements and Test Cases

Requirements were implemented first and can be seen in Chapter 4, Section 4.2.3. Then test cases to the requirements were made to be able to validate that the requirements have been fulfilled.

3.3 Simulation

A simulation was created that tested different types of policies of detecting phishing emails. The purpose was to gather data about the policies to be able to make a logical choice of which policy to use. The simulation evaluated the desensitization risks and when the software did not nudge against phishing email. The simulation was written in the programming language Java and has the structure as the JSP-diagram shows on the next page, the Figure 3.4.

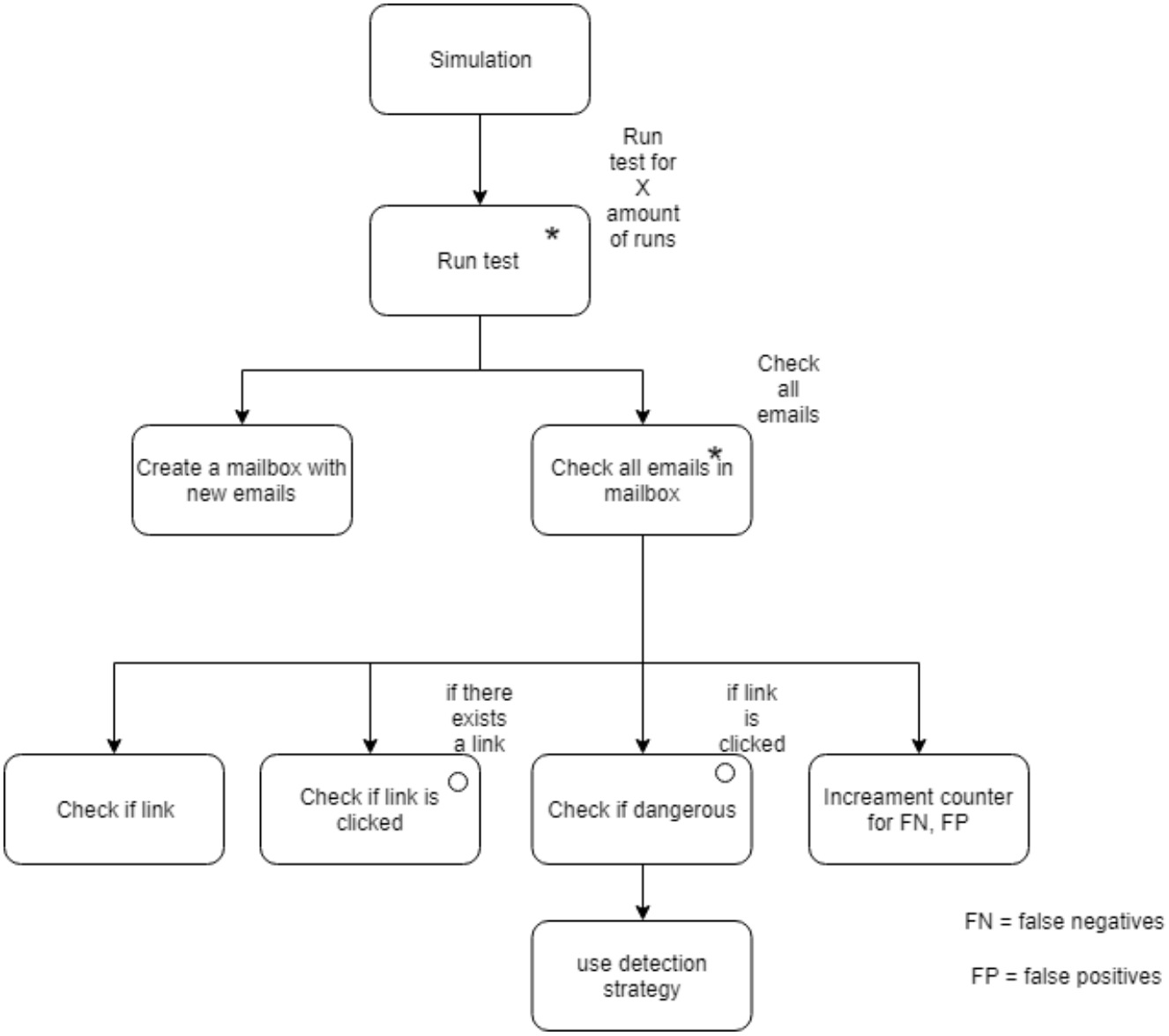

Figure 3.4: JSP-diagram of the Simulation model

The JSP-diagram, Figure 3.4, shows the simulation's model. The model shows that the program should create a mail box and go through them all, an X amount of times. The box "Create a mailbox with new emails" can be viewed in detail in the next JSP-diagram, the Figure 3.5. When the program is checking an email, it should check what type the email is and run probability tests to see if the program does certain actions and count successful and failed attempts.

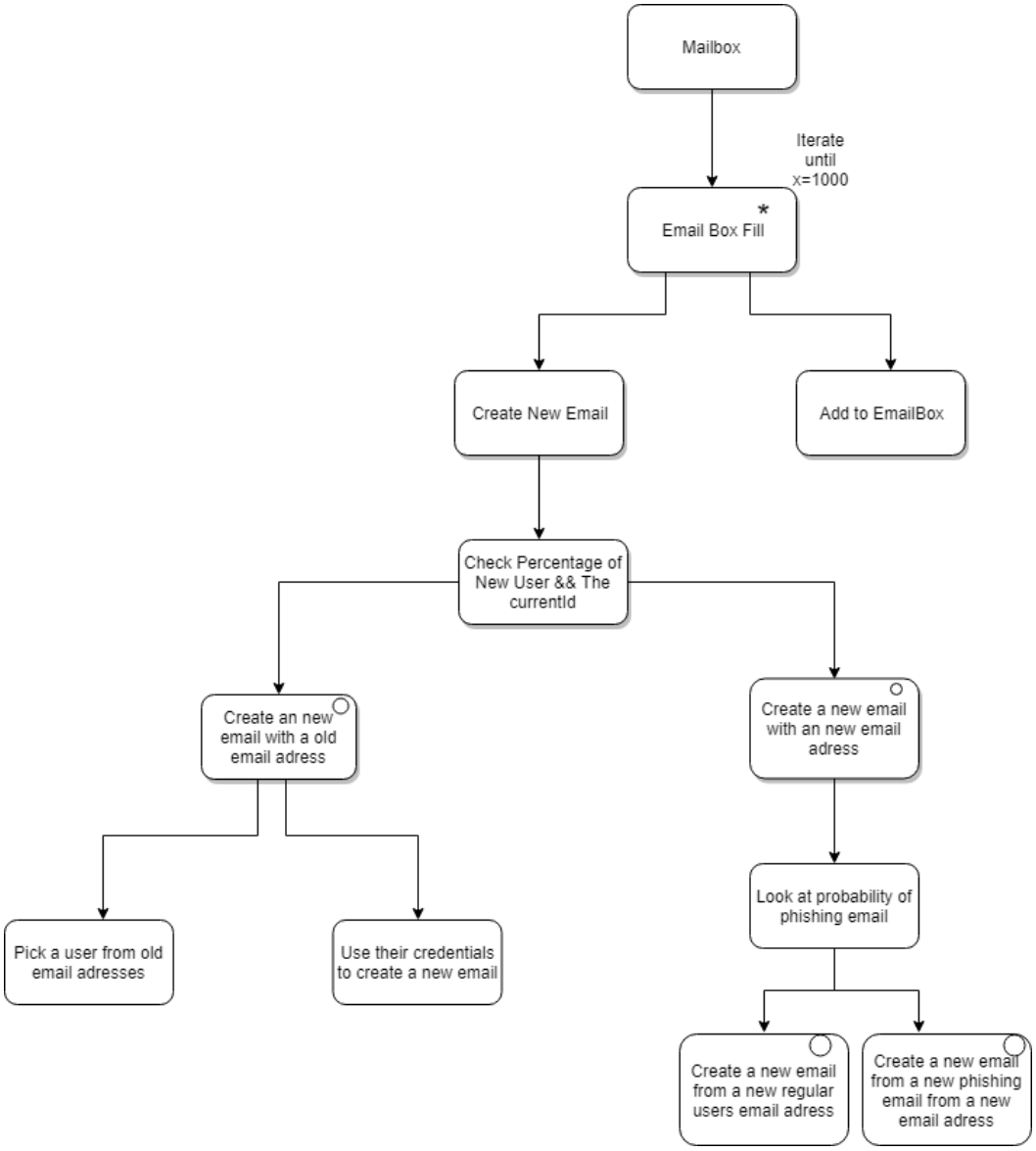

Figure 3.5: JSP-diagram of how the emails inbox is built.

The previous JSP-diagram, the Figure 3.5, shows the model of the process of creating a mailbox. The size of the email box was decided to be 1000. It creates an email with different attributes depending on probability variables. Then it adds it to the mailbox, and when the mailbox is full it continues with the simulation as shown in the first JSP-diagram, the Figure 3.4.

3.4 Survey

A qualitative survey was made to test the user experience. There were three objectives that were evaluated:

- Possible positive behavioural changes.
- Possibility of desensitization.
- Reliance on the software, meaning that the users trusts the software even if the software is incorrect.

The survey was done in a focus group setting. The participants first answered questions about their background regarding phishing emails and then the participants used the software. They viewed a folder with emails, where some were phishing emails. They were told to go through the emails and click the links. Then questions about the experience were asked. The sample size was small, due to the time constraint. The people interviewing the participant were the authors. The survey participants consisted of relatives, friends and family members.

Annoyance can lead to desensitization, as people will try to avoid negative feelings. Therefore, annoyance was used to as a factor to predict possible desensitization.

The questions were originally asked in Swedish and can be viewed in the appendix, Section A.3.

The questions asked were the following:

- Do you know what phishing is?
- Have you ever fallen for phishing?
- Do you think that this nudge could change your behaviour?
- This is meant to be used in a company setting. Where being careful is important. If you saw this warning how would it impact your behaviour?
- Do you think that the nudge would be annoying if it appeared when you opened an email? Answer how annoying it would be from a scale 0 to 10,

were 0 means not annoying and 10 means very annoying.

- How often would the nudge have to occur to be considered annoying?
- In a company setting, would you think that the nudge would be annoying if it occurred too often? If so, why and how would you minimize the impact?
- How did you think the nudge looked? Are there design choices you would want to change?
- In a company setting, if the nudge did not occur, would you still be critical of the emails you viewed?
- If there is anything else you would like to mention, please do!

Possible answers for some questions were ”Yes”, ”No”, ”Maybe” or ”I do not know”. While the others had a section were people could write a more detailed answer.

4 Design, Implementation and Tests

In this chapter everything related to design and implementation are presented. The different design choices, the argument for each choice and how the progression went are presented here. The software design can be found in the Section 4.1.1 and the nudge design in the Section 4.1.2. The implementation of the tests can be found in the Section 4.2. The simulation progress in the Section 4.2.1, the survey progress in the Section 4.2.2 and the creation of test cases in the Section 4.2.3.

4.1 Design Decisions

In this section, one finds the practical descriptions of how the methods were applied to the software design and the nudge design. There are also notes about the progress, how it went and which mistakes that were made.

4.1.1 Software design

After learning about the general structure of a Chrome extension, the implementation could begin. The initial software was a simple extension that redirected all the links in a webpage to another link. Webpages are built with HyperText Markup Language (HTML), and HTML uses tags to define certain text and give it attributes. The anchor tag is used in almost all phishing emails to redirect the user to a different webpage than the one intended. An important attribute of the anchor tag is the Hypertext REFerence (hrefs) attribute, which creates a link to another page. So, the first small test extension changed all hrefs to the same link. This was moderately simple since the only thing that was need was a loop to change all the anchor elements on a webpage.

To run the extension in an optimal manner in the email service Gmail, a decision was made to use an unofficial API, called Gmail.js. This is an API specifically made for Chrome extensions. There were some issues with the initial start-up of this API, since most of the API returned a jQuery object, which gave an error. jQuery

is a JavaScript library that makes it easier to use JavaScript on a website.The program started to use jQuery as it was required if one wanted to use the API. An interesting feature from the Gmail.js API was the view_email function. This enabled the plugin to run a specific function only when an email was opened which, was perfect for the software.

The next task was to display a dialog instead of a redirecting to new webpage when clicking a link. First a click listener was added, which the Gmail.js API had a function for. An issue was that the href attribute redirected the user to another page when the dialog was displayed. The solution was to prevent the redirecting. This was done by making the href actionless. This displayed the dialog, instead of the user being redirected to another page. The following code made the href void, and therefore actionless.

```
emailBody.find('a').each(function() {
    if(typeof($(this).attr('href'))!='undefined'){
        $(this).attr('href',"javascript:void(0);")
        $(this).attr('data-saferedirecturl',"javascript:void(0);")
     }
});
```

The integration between the dialog and the link detection had some issues with the JavaScript code that was embedded in the code of the dialog. One was forced to take the dialog code out and put all the JavaScript code in the extension's code instead. After that the code worked fine and the software was able to pass all the test cases related to the basic functions of software.

The filters were implemented as the definitions were described in the Method, Section 3.2.2.

4.1.2 The Nudge Design

An HTML dialog element was used to show the pop-up. This dialog had the possible nudge effects in it. Another alternative that was considered was a modal element as it is compatible with more browsers, but the HTML dialog element could easily blur the rest of the screen and make the dialog stand out. This lead

to that the HTML dialog element was used. The first dialog only contained a button that could close the dialog when pressed. This was a very basic dialog that completed one of the test cases, but was something to continue to work on. The dialog had the following code:

```
<dialog id="dialog">
  <p>Hi, I'm a dialog!</p>
  <button id="close">Okay</button>
</dialog>
<button id="show">Show Dialog</button>
```

Instead of expanding the code within the dialog element, the rest of the development was done in a separated HTML file. When the HTML file was completed and looked like the draft model as described in the Method's Section 3.2.3, one could easy convert it to a dialog element again and merge it with the plugin program.

One choice when creating the form was to use radio buttons as the user should only be able to choose one alternative. JavaScript code was made to run in the background. The JavaScript code fetched the input from the form and alerted different outputs depending on which alternative that was selected. If the user chooses the ”Spam or Phishing” alternative, the alert ”close dialog” will show, here a close dialog function were later implemented. If one picks any other alternative, it will continue to the next page. The JavaScript described was the following:

```
<script>
    // Fetch the form
    var radios = document.getElementsByName('fromwho');

    function chechWhatUserChoose(){
        for (var i = 0, length = radios.length; i < length; i++)
        {
            if (radios[i].checked)
            {
                if(radios[i].value!="spamorphishing")
                {
```

```
                    alert("Next page");
                }else{
                    alert("close dialog");
                }
        // only one radio can be logically checked, don't check the rest
            break;
            }
        }
    }
</script>
```

When the first page had its basic functions done, the next step was to implement the second page. Different division HTML tags were used to toggle between different pages. One could hide one page and show the other, to create a page swap using HTML division tags.

Last part was to make it look more like the model described in the Method's Section 3.2.3, with an improvement to the aesthetics. Cascading Style Sheets (CSS) was added within the HTML file, not a separate file, to make it easier to merge with the rest of the program later. The HTML pages looked like the following images after improving the aesthetics, Figures 4.1 and 4.2.

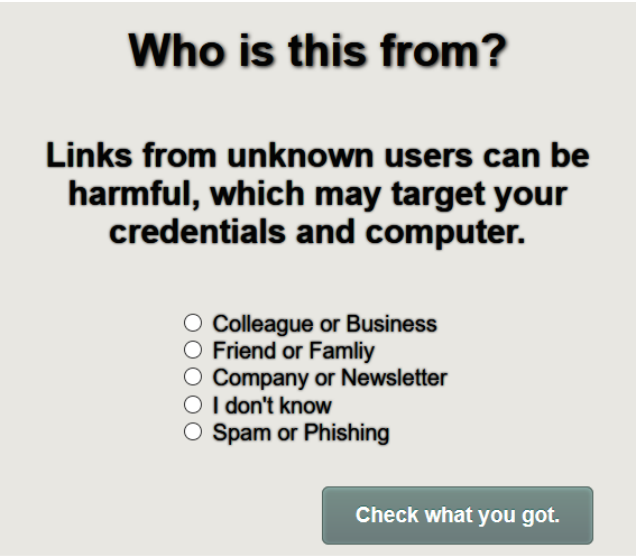

Figure 4.1: Draft dialog page one, with added CSS.

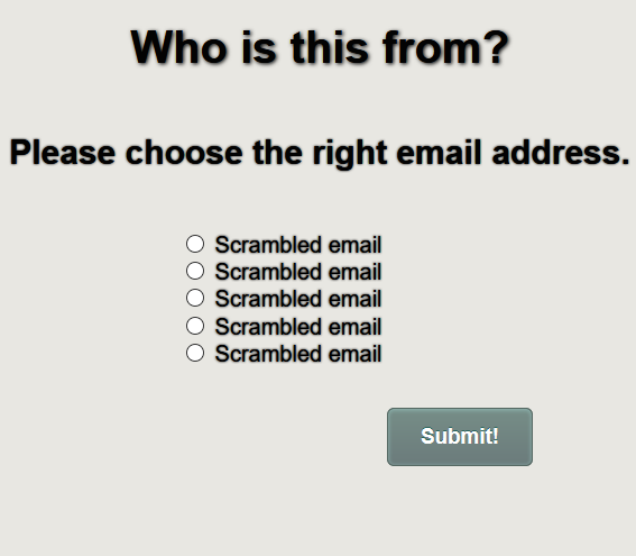

Figure 4.2: Draft dialog page two. This picture shows that the scramble email has not been completely done yet.

The last part was to implement the email generator, which was done by a scrambler. More about the scrambler in the next Section 4.1.3.

Something the model did not have, but that was added, was direct feedback to the user after a choice was made. Basic feedback that told the users if they choose the right or wrong answer. The following picture, Figure 4.3, is taken from the plugin after one chooses the right option.

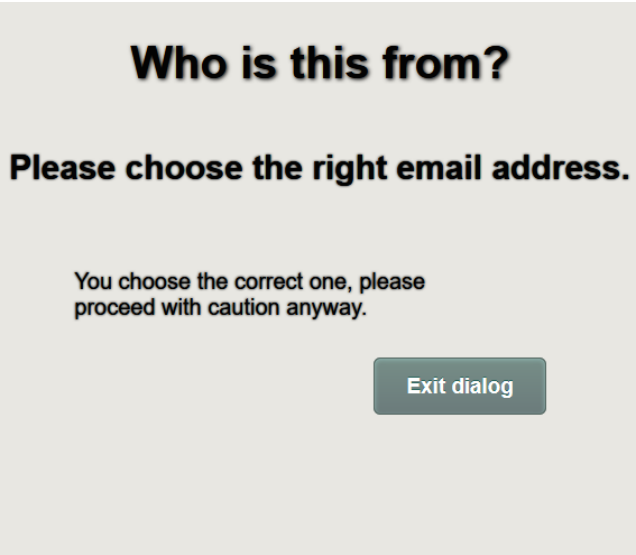

Figure 4.3: Dialog page ”Correct answer”. This picture is shown after the right alternative has been picked as feedback.

4.1.3 Scrambler

The continuation was to develop the scrambler script in the dialog file, before the merge with the rest of the plugin. The basic functions of the scrambler were to find the email that should be scrambled and the scrambling sequence. The function to find the correct email was delayed until after the merge of the dialog file and the rest of the plugin file. Instead, development began with the scramble function. The scrambler therefore was first made with a test email, test.prov@mail.com. A separation between the ”@” symbol could be made with the JavaScript function split(). And then the split function continued to split, but with ”.” as a separator instead. See following code.

```
<script>
    var emailToBeScramble = "test.prov@mail.com";
    var arraysplit1 = emailToBeScramble.split("@");
    var frontpart = arraysplit1[0];
    var backpart = arraysplit1[1];
    var frontsplit1 = frontpart.split(".");
    var backsplit1 = backpart.split(".");
</script>
```

The next part was to implement the creation of the look-alike email addresses. The construction of the addresses was done by combining the parts from the split in different ways. Some of the created email addresses can be found in the Appendix, Section A.4.

4.2 Implementation of tests

In this section the process of the implementation of the simulation and the questionnaire form is presented, as well as the creation of the requirements and the test cases.

4.2.1 Simulation

The program had one main class and one email representing class. The main class contained the main function which ran as the JSP-diagram, which can be seen in Figure 3.4 in Chapter 3. Input data was set in the initiation phase. The simulation had variables that controlled the accuracy of how good it used certain policies. Some policies could be user-dependent, and if so, the simulation needed to make judgments on its own. Examples on variables that could be changed were the percentage of correct whitelisting that occurred by the simulation, or how much the simulation clicked on phishing and non-phishing links, as well as if the simulation learned to click less on phishing links. Some of the polices depended more on the variables that control the structure of the mailbox. For example, all the polices depended on the variable that controlled the percentage of phishing emails in the mailbox and on the percentage of reoccurring emails.

The email class contained three attribute variables. An ID, which indicated who the email was from. A Boolean that represented if the email contained a link or attachment. And a different Boolean, which if true, represented that an email was a phishing email. The JSP-diagram which can be seen in Figure 3.5 in Chapter 3, shows how the email inbox was built.

The test ran with the mailbox size of 1000 emails. Different counters were implemented to see how the strategies performed overtime. The different counters counted in different intervals, which were between “0 and 100”, “0 and 500”, “500 and 1000”, and “0 and 1000”. Some counters showed how many times the nudge did not trigger for a phishing email, which was defined as a false positive event, and some showed how many times the nudge triggered for a non-phishing email, which was defined as a false negative event. ”False” was defined as when the simulation did not trigger a nudge at the appropriate time, while ”True” when it did trigger at the appropriate time. ”Positive” was defined as when an email was a phishing email, while ”Negative” means that it was not a phishing email. False positives were bad cases, because in the case of a false positive event, the users will not see the nudge and will instead be redirected to the hacker’s site. The false negatives were connected to the desensitization problem, as in these cases, a nudge is not needed. Many nudges may cause unnecessary annoyance in these

false negatives situations. For more explanation about true and false positive or negatives, see the matrix Table 5.2 in the result's Section 5.2.

The simulation ran with the same inputs a million times and then the means for each counter was calculated. By calculating the means, one gets results closer to the expected value, accordingly to the probability theory law of large numbers [15]. Each strategy method depended on its relevant parameters, such as how many recurring emails or friendly contacts there were. All the different strategies commonly depended on factors like the click percentage or the phishing email distribution. The distribution of phishing emails were 1/58, based on a Symantec threat report [39]. This data was specifically for companies of a greater size. The average click rate of a phishing email was around 9% and the average click rate for marketing emails was around 1.9% [35][31]. When setting the parameters for the simulation the worst-case was assumed. The link click percentage for a phishing email was for example 100%.

The creation of the mailbox was one of the biggest factors to the test's result, as the performance of the strategies heavily depend on it. The goal was to mimic a real-world scenario as much as possible. Overall, the simulation was designed to represent the worst-case scenario. A great example of this is that the simulation clicked on every phishing email and the simulation did not learn overtime to click on less phishing links. Decisions about different distributions of emails was assumed to be around 80% reoccurring contacts and 20% new emails. The first ten emails that were created were friendly known contacts. To mimic that 1/58 of emails of the total inbox were phishing emails, the percentage that a new email was a phishing one was set to 8,709%. With that value, 1/58 of the email box size of 1000 were phishing ones, on average. The click percentage of non-phishing emails was set to 70%. Another percentage which was assumed was that a user guesses right with 95% accuracy when whitelisting. It was also assumed that the user had sent an email or started a conversation with 10% of the inbox population. Confidence intervals for the total percentages were created.

After the results were gathered, they were displayed as graphs were the y-axis represented the percentage of the events false negatives or positives, and where the x-axis was the different interval counters. These graphs showed how the

different strategies performed overtime. A graph was created for each detection strategies. Two graphs were made that showed how the strategies performed compared to each other. One for the event of false positives and one for the false negatives. With these graphs one could make comparisons between the strategies easier.

A code snippet of the simulation checking the emails can be found in the Appendix's Section A.2. The snippet has been modified for viewing, which means that many counters have been removed from it. The raw data from the simulation runs can be seen in the Appendix's Section A.1.

4.2.2 Survey

As mentioned in the Method's Section 3.4, the survey was done in focus groups. The interview took place at different locations, depending on the participant's needs. Often at their house, but sometimes at a KTH campus. The interviews started with a small introduction to the problems of phishing. Then, an explanation of the purpose and the goal of the software was given. After that, a short demonstration of the created nudge was shown. Lastly, the participants were asked to answer a survey. They could ask questions for clarification. Important comments and notes were documented. The survey answers were collected through survey monkey. The data that was collected was used to plot pie charts using Microsoft's Excel.

4.2.3 Test Cases and Requirements

The requirements for the implementation are presented in the list A.1, which can be found in the Appendix's Section A.5. The list was sorted into levels, which represent the scope of the requirement. When a lower level was working correct, all the higher leveled ones were working as intended as well. By splitting up the requirements one could easily see what needed to be done to accomplish a requirement. Each specific requirement had a description of what should be met, but also the connected test cases to it. The test cases could validate that a requirement had been met. All the test cases are described in the list A.2,

which can be found in the Appendix A.5. Here below, one finds one example of a requirement item shown in the Table 4.1, and one test case shown in the Table 4.2.

Requirement Title	1 - Installation works
Level	0
Description	Must be able to install and have the software program work as intended, which means following our JSP-diagram. The program works as intended if the installation works correctly and all the other requirements are met.
Test cases	• 0.0 - Installation • 0.1 - Complete run 1 • 0.2 - Complete run 2

Table 4.1: Example of one requirement item

Title	0.0 Install
Description	Tests if the program can be installed and ran.
Precondition	The user has access to the code.
Assumption	The user has a browser supported for plugins.
Step	• 1.Npm runs successfully • 2. Npm builds without warning. • 3. Upload directory with manifest.json to chrome • 4. Go to Gmail on browser • 5. View Console to check if the extension runs accordingly
Expected Results	If the program runs the installation worked.

Table 4.2: One example of the test cases

5 Result

This chapter presents all the results from the survey and the simulation. The survey's results can be found in the Section 5.1, which contains the important comments that were documented during the interviews, Section 5.1.1. Section 5.2 contains results related to the simulation. Graphs that describes different interesting events are shown the Section 5.2.

5.1 Survey Results

The survey's results are presented by three pie charts, where each chart relates to a specific question. The important comments, Section 5.1.1, includes relevant comments that describes the possible mentalities and the general features that the survey participants wanted. The following pictures have descriptions that includes the question that is related to them, as well as what the results are indicating.

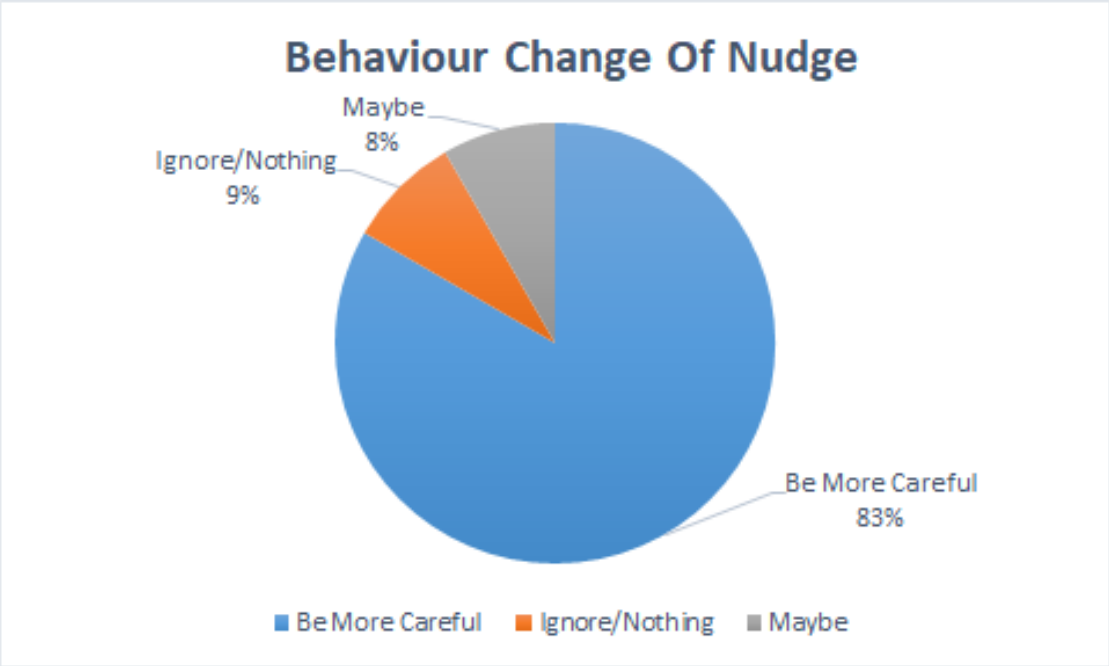

Figure 5.1: The figure shows the answers to the following question: "*This is meant to be used in a company setting where it is important to be careful. If you saw this warning, how do you think it would impact your behaviour?*". The answers are categorized into being more careful, ignore/nothing and maybe. This graph gives an indication of possible behavioral changes due to the nudge.
Note: The people that answered ignore had less experience with phishing emails.

Figure 5.2: The figure shows the answers to the following question: *"In a company setting, would you think that the nudge would be annoying if it occurred too often? If so, why and how would you minimize the impact?"*
This figure gives an indication of the issues with false negatives and how they might increase desensitization of the nudge.

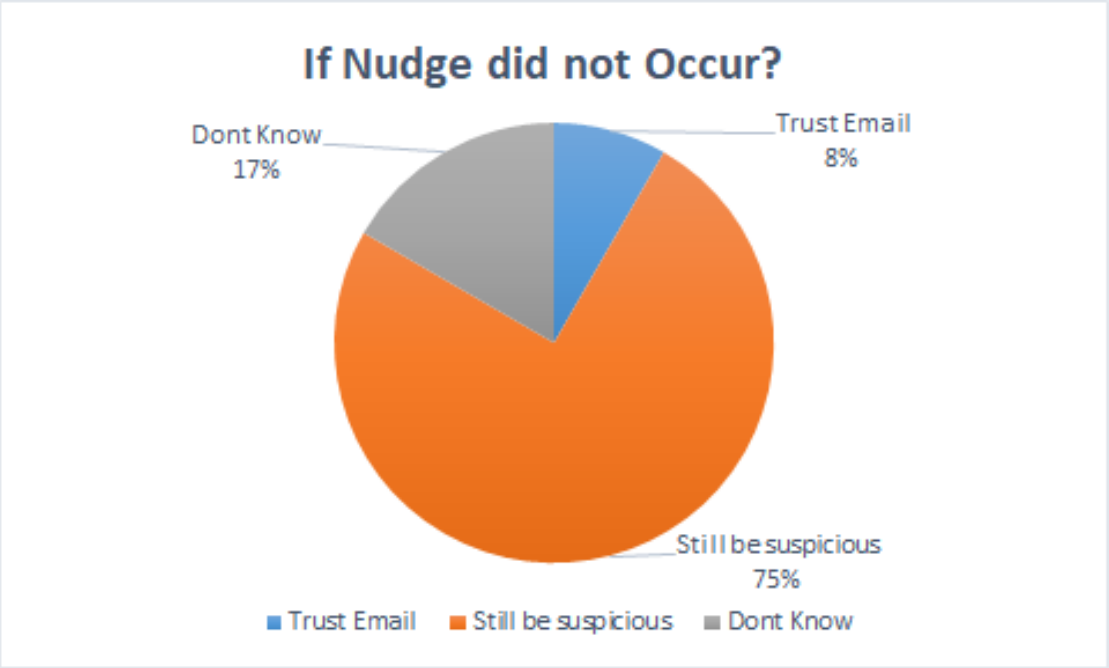

Figure 5.3: The figure shows the answers to the following question: *"In a company setting, if the nudge did not occur would you still be critical of the emails you have?"*
This figure gives a possible indication of the risk of false positives.
Note: Most participant that picked "Trust Email" and "Don't Know" were participants with less experience with phishing emails.

5.1.1 Important Comments

This section contains comments that are relevant for the evaluation of the nudge. These following comments were specially interesting and constructive.

- *"Have an introduction on the topic so that people can get context or have a tutorial so that people can understand the purpose of the nudge."*
- *"The application caused confusion and therefore I would not use it."*
- *"Add more text to reduce confusion and do make the warning clearer."*
- *"It would be nice if there was a detection without the need for me to interact."*
- *"The nudge is a bit too aggressive since it makes everything in the background dark, which made it seem intimating at first. One suggestion I would make is to have a smaller notification on the link. This would give*

the users a nudged even before clicking the link."

5.2 Simulation Results

The simulation's results are presented as bar charts, which represents the average values from the simulation of one million runs. The raw data of the inputs and outputs can be found the Appendix, Section A.1. The false positive and false negative were the most significant events and were defined as in the matrix below. Similar to other tests have done, as Google defines it:

"A true positive is an outcome where the model correctly predicts the positive class. Similarly, a true negative is an outcome where the model correctly predicts the negative class. A false positive is an outcome where the model incorrectly predicts the positive class. And a false negative is an outcome where the model incorrectly predicts the negative class." [8]

In the simulation's case, the positive class was "it is a phishing email" and negative class was "it is not a phishing email". If true, the simulation's outcome from an email was correct, and false if not. See the matrix (the Table 5.2) below for more clarification.

True Positive (TP): • Reality: A phishing email. • The nudge does trigger. • Outcome: Good outcome.	False Positive (FP): • Reality: A phishing email. • The nudge does not trigger. • Outcome: Worst outcome. User will continue with the bad link click.
True Negative (TN): • Reality: Not a phishing email. • The nudge does not trigger. • Outcome: Good outcome.	False Negative (FN): • Reality: Not a phishing email. • The nudge does trigger. • Outcome: Bad outcome. Might cause unnecessary annoyance.

Table 5.1: Definitions shown with a 2x2 confusion matrix

The following results below each have four series, which are ”FN-allmails”, ”FP-allmails”, ”FN-linkclicks” and ”FP-linkclicks”. FN stands for false negatives and FP for false positives. The ”allmails”-tag presents the percentage of the events FN or FP occurrence, where the total is every mail in the inbox. While on the other hand, the ”linkclicks”-tag presents the percentage of the times the events occurred but only those cases where the simulation has clicked on a link, which might be more interesting as it is more relevant when judging the strategies. These series are represented as bars, where the y-axis represents the percentage of the corresponding event, FN or FP. The complement event to FN and FP would be true negative or true positive. Down below are the results from the double strategy, see the Figure 5.4. An explanation about the strategies, can be found in the Method, Section 3.2.2. On the following pages the results of the whitelist strategy (Figure 5.5), the sentlist strategy (Figure 5.6) and the combination of both the whitelist and the sentlist strategy (Figure 5.7) are presented.

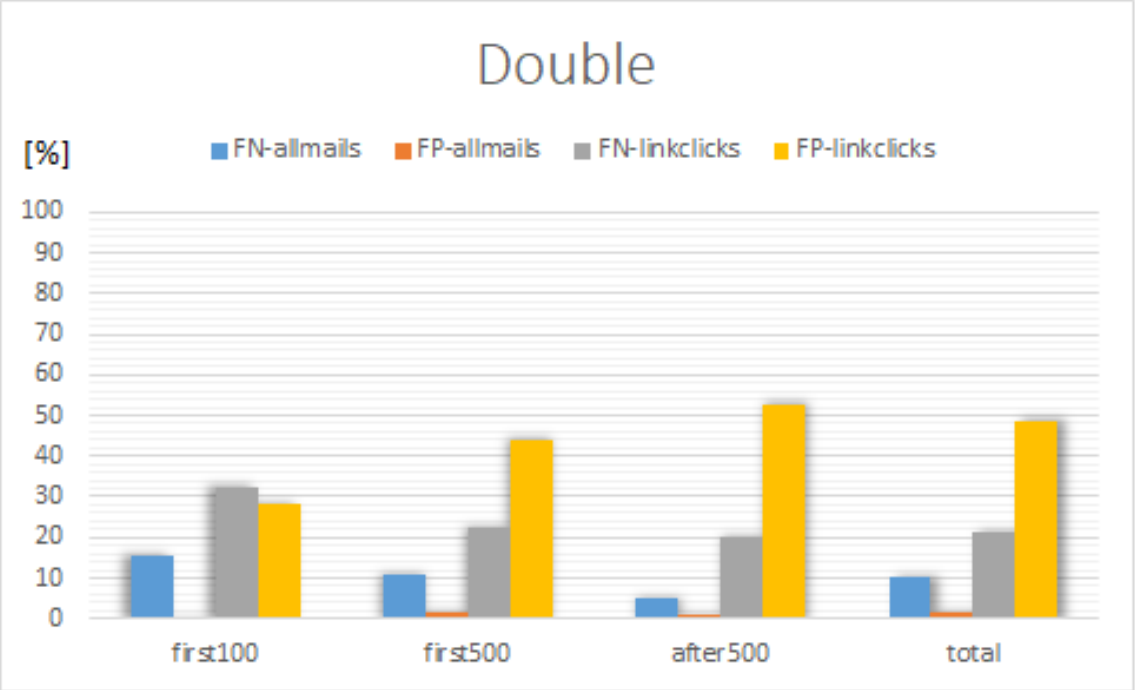

Figure 5.4: Results from the Double Strategy

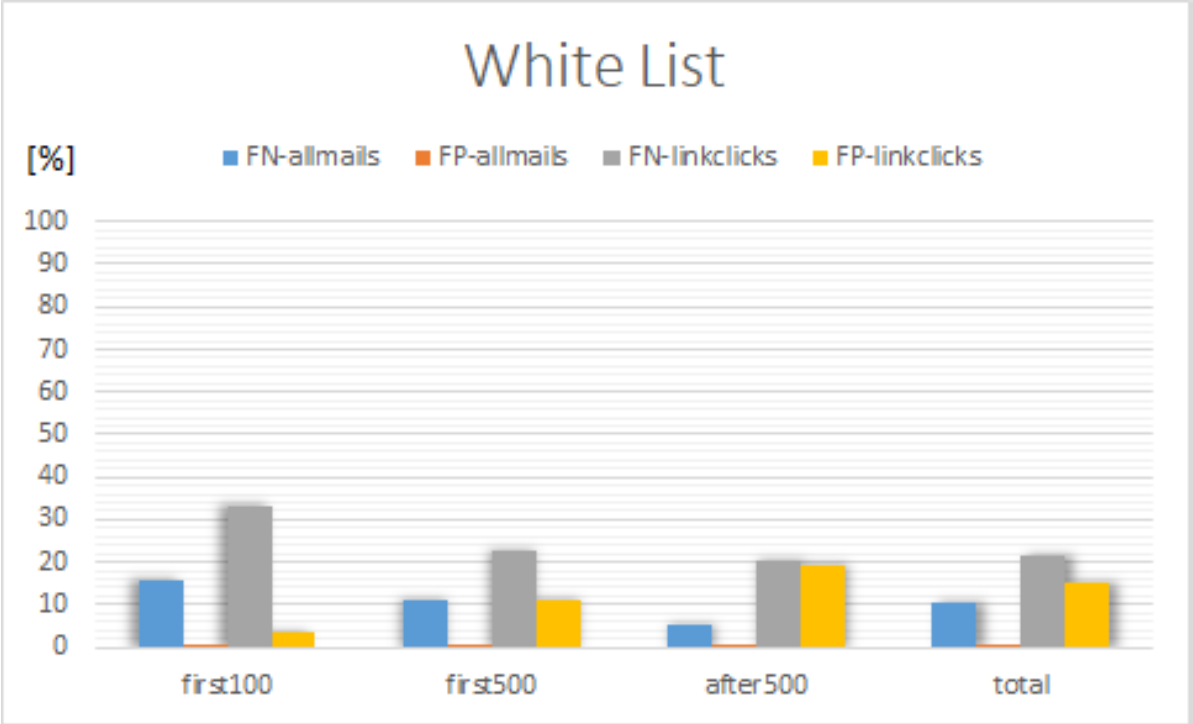

Figure 5.5: Results from the Whitelist Strategy

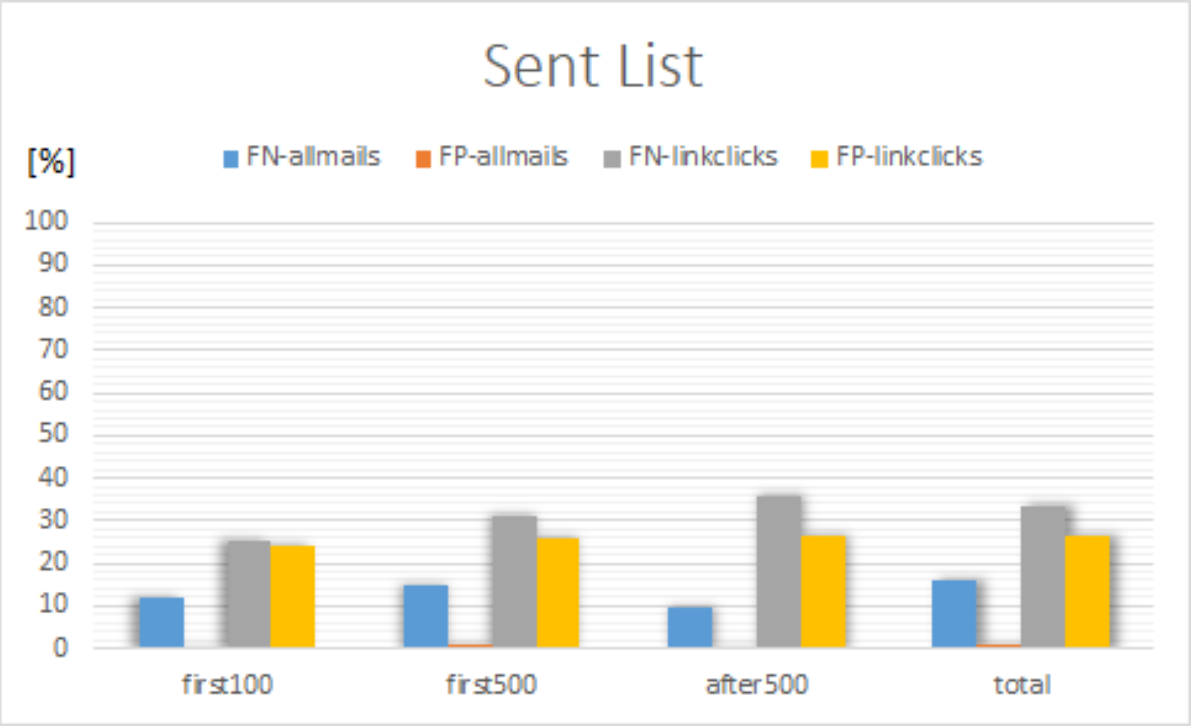

Figure 5.6: Results from the Sentlist Strategy

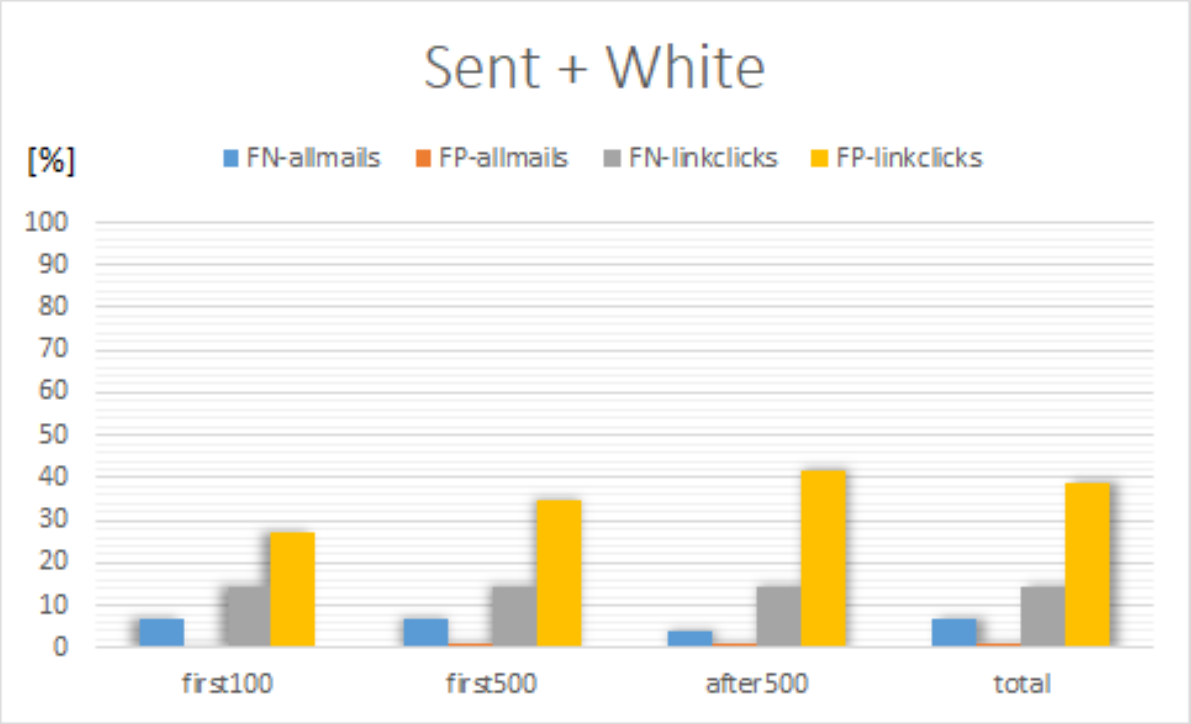

Figure 5.7: Results from the combination of the Whitelist and the Sentlist Strategy

The following two bar charts (Figures 5.8 and 5.9) below represent the previous series' FP-linksclicks and FN-linksclicks, which are now the only events that are displayed on the y-axis. The bars are now representing each strategy instead. The data is used from the previous bar charts, but it is now easier to compare the strategies against each other. Confidence intervals were created for the total interval for false positive and negative. The intervals were so small as the program ran many times, that it would not make sense to present them in the bar charts. For the double strategy, the result was 21.347% false positive within the interval [0.21347415665967437, 0.21347488236920112], with a confidence level of 95%. The first change started at the seventh decimal.

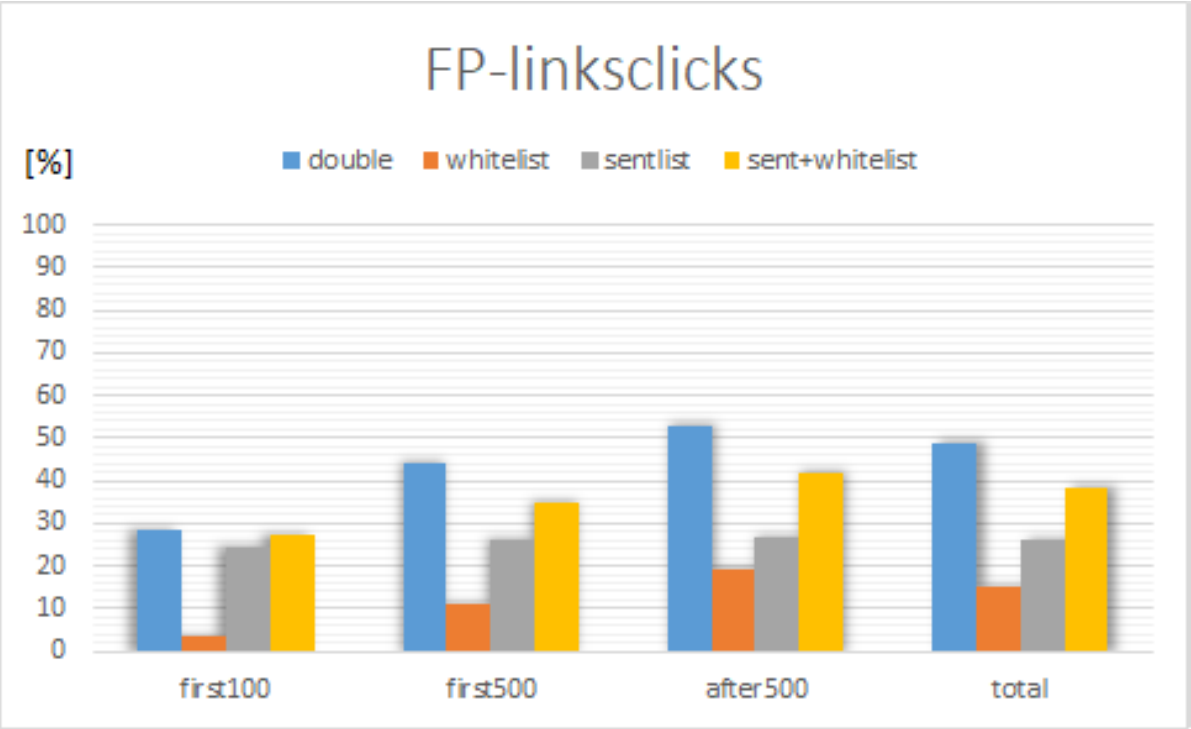

Figure 5.8: Results from the event False Positive

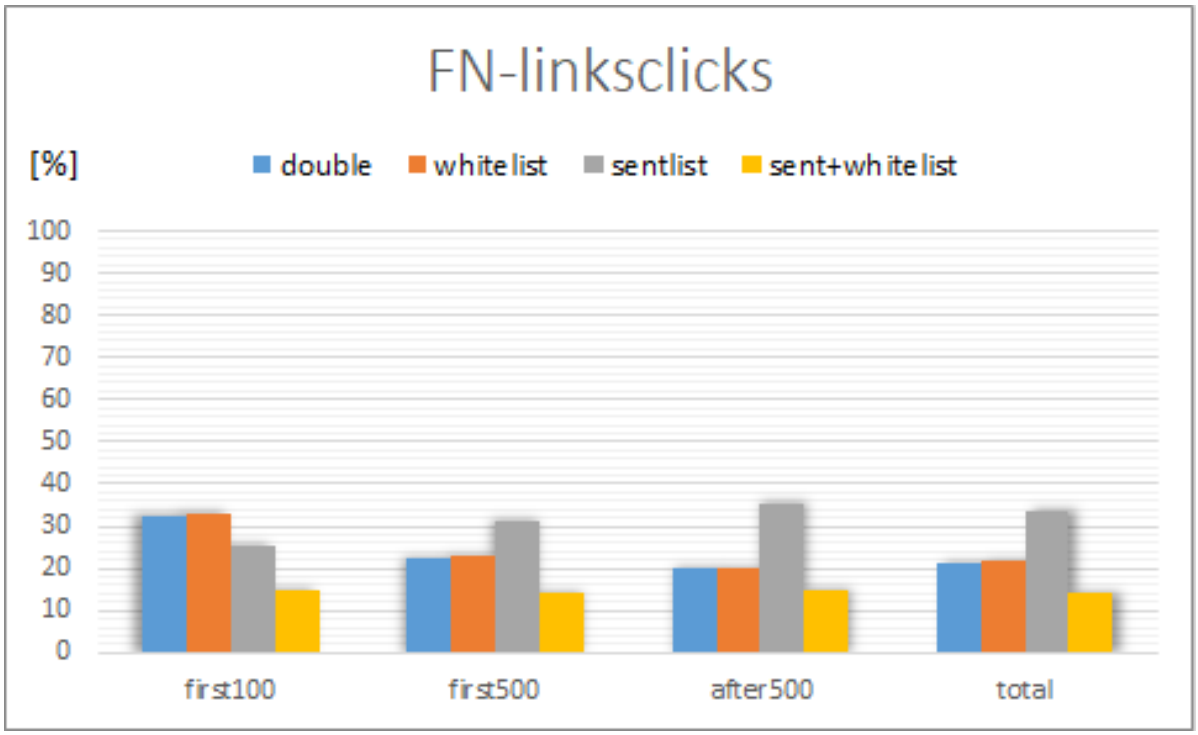

Figure 5.9: Results from the event False Negative

6 Discussion

This chapter includes a discussion about the results and compares it to the theoretical background and the research question. The discussion is first divided into a nudge specific part and a simulation part. The nudge specific part, Section 6.1, relates to the result of the survey, while the simulation part, Section 6.2, discusses the results of the simulation with a great focus on the false positive and false negative events. The discussion ends with the Section 6.3, which presents possible problems of the given solution and suggests improvements.

6.1 Nudge Discussion

The discussion consists of two parts, one where the result is analyzed and discussed, and the other part presents possible nudge improvements.

6.1.1 Survey from Focus Group

Based on the answers shown in the Figure 5.1 in the Section 5.1 of the Results, the nudge seemed to have a positive change in the behaviour on most participants in the survey. Though, as noted in the important comments, Section 5.1.1, some participants deemed the nudge as aggressive. Arguably, the nudge might have been too aggressive since it stops all the email-client functions. This can be perceived as forceful rather than suggestive. One suggestion from the important comments was that the nudge could have been done by having a smaller pop-up being displayed when hovering over a link. The pop up would still warn people about the dangers of phishing but would make the nudge less aggressive. This suggestion could be implemented, but it might make the nudge less effective. Examples of less aggressive nudges are described in the Theoretical Background, Section 2.6.4.

The answers shown in the Figure 5.2 shows that 50% of the participants considered the nudge to be annoying. This indicates that 50% of the participants could be desensitized by the suggested nudge. As stated in the important

comments, the participants viewed the nudge as aggressive, which may be the reason for 50% of the participants deeming the nudge as annoying. This suggests that desensitization is a problem in the suggested nudge. Making the nudge less aggressive would seem to be an important improvement.

The answers shown in Figure 5.3 shows that 75% of the participants would still be suspicious of a link even if the nudge did not occur. An explanation for this could be that most of the participants had some knowledge about phishing. No conclusion about false positives can be drawn from the Figure 5.3, since the figure only shows how participants think they would act.

One can be skeptical of the data that was gathered by the survey, since the sample size was small. Most participants were young adults and therefore the data cannot be generalized to an entire population. Overall, there are indications that the nudge could change the behaviour of some people.

6.1.2 Nudge Improvements

A nudge effect that was not used was the social norm effect. In the Theoretical Background 2 the concept was introduced. The social norm effect could lead to beneficial outcomes. The effects that were used were modus ponens sentences, framing effects, elaborations and reminding the user frequently of the nudge.

The choice of having a dialog implementation might have been too aggressive in hindsight, but to verify that, one would have to test the plugin against one that is less aggressive. A less aggressive nudge could have been a color-coding one, as described in the Theoretical Background, Section 2.6.4. Tests of the plugin are planned to be made, but only to see how it performs and it will not be compared to other types of nudges. To find out more about the future work, see the Chapter 8.

6.2 Simulation Discussion

The simulation was used to evaluate the filtering methods presented in the Section 3.2.2. The worst events that could happen during the simulation were false positives. In a real-life scenario of a false positive event, a user might assume that a link is safe even though a link might be unsafe. False positives should therefore never occur. Another factor to consider is the false negatives. False negatives can cause desensitization to the effect of the nudge. Reducing false negatives is therefore preferable.

6.2.1 Simulation Evaluation

The double strategy is the worst based on the results from the simulation. It had the highest number of false positives, based on the Figure 5.8. The number of false negatives were low as one can see in the Figure 5.9, but because of the high number of false positives, the double strategy is still considered to be the worst. There is a possibility to improve the solution, which is discussed more in the next Section 6.2.2.

The sentlist had the second lowest number of false positives, based on the Figure 5.8. The result for the sentlist was better than the double strategy because of its lower number of false positives. Though, in the Figure 5.9, the number of false negatives were the highest, which indicates possible issues with desensitization. Sentlist is overall better than the double message strategy due to the lower number of false positives.

The sentlist + whitelist strategy had the lowest number of false negatives, based on the Figure 5.9. However, the number of false positives were the second highest, based on the Figure 5.8. The solution is therefore worse than sentlist because of the high number of false positives.

The whitelist is the best solution overall. The solution has the lowest number of false positives while also maintaining a low number of false negatives. This can be observed in the Figures 5.8 and 5.9.

A deeper dive into the possible reasons for these results can be found in the next

Section 6.2.2.

6.2.2 Discussion on each List-based filter

Phishers can send phishing emails multiple times. If they would, it could create a problem for the double message strategy, as it would lead to incorrect whitelisting. A solution to the double message strategy's problem is to have a timeout on the whitelisted email addresses, meaning that after an X amount of time a whitelisted email address will no longer be whitelisted. This would reduce the number of false positives. This solution would still have a problem if a phisher sends another email before the email address gets timed out. One can decided that the double strategy was the worst of the list-based strategies, due to the possible issue with the solution and the high number of false positives compared to the other strategies.

Sentlist has some inherent issues. If phishers starts an conversation with the users before trying to lure them, it would assure that phishers would become whitelisted. When the phishers are whitelisted, the nudge will not trigger and the solution will become no longer be effective. Another issue is that some emails that the users will receive are not from addresses that the users would send emails to. For example, newsletter emails. The users will never send an reply to these newsletters, but newsletters do contain links. This leads to issues with increased levels of desensitization, since the number of false negatives would increase.

The Whitelist is arguably the best solution, but the solution is heavily dependent on the experience of the user. Users with less experience might whitelist a phisher. However, the nudge enables people to make safer choices when it comes to whitelisting. The nudge will ask the users if they can recognize an email address, then ask if the users should whitelist the email address. The nudge will warn them if the user answers incorrectly. Users might reconsider more before choosing to whitelist an email address if they have been warned. The whitelist strategy helps the users to make better choices and makes them more safe from these attacks.

Sentlist + whitelist has certain problems. The assumption was that the benefits of

sentlist and whitelist would overlap. Instead, the shortcomings of each solution became more apparent. When looking at the Figure 5.9, one can observe that there is a higher number of false positives than the separate strategies on their own. One could guess that these problems occurred because the combination of the sentlist and the whitelist is dependent on two different list-based strategies. The number of false negatives is low in the sentlist + whitelist, based on the Figure 5.8. In conclusion, the sentlist + whitelist is better than double message strategy but worse than sentlist due to the high number of false positives occurring.

6.3 Problems with the Suggested Solution

Email address spoofing is a problem for the implemented software. The software is unable to differentiate between a spoofed email address and a real email address. Therefore, the nudge will not trigger if a phisher spoofs an email address that is already whitelisted. The user will then assume that the email is safe, even though it is not. The software will be able to deal with email name spoofing, since it still displays the actual email address of the user. Generally, the email name and the email address will have similarities, if they do not have any, the user will be able to tell the difference with the program. Email spoofing was a problem that was not considered at first, but the solution still has merit since it will deal with the most common forms of phishing attacks.

A link-based nudge solution could be implemented instead of the suggested nudge. This solution would make the users aware of the destination of the link rather than focusing on the sender of the email. Still, the problem of how to reduce desensitization would exist. The link-based nudge is introduced in the Theoretical Background, Section 2.6.4.

7 Conclusion

As reminder, the research question of the thesis was the following:
"How to implement a nudge against phishing with regard to possible desensitization?"

To answer that question, one could implement a nudge using an email extension. The email extension could display a dialog that would have all the features of a nudge. The dialog could trigger at appropriate times, which would be when someone clicks a link of a phishing email. Asking questions regarding the email sender's address raises the awareness of possible phishing attempts. Based on the survey, it can be assessed that the nudge can possibly have an impact on the behaviour of the participants. Regarding possible desensitization the whitelist is the best strategy for the nudge. It had the least number of false positives occurring, whilst having a low number of false negatives happening. This conclusion was drawn from simulation results.

The plugin that was created will hopefully lower the number of successful phishing attempts on the users of the plugin.

8 Future Work

The future for the program is that it will be tested by the Danish authority Ervhervsstyrelsen in a Randomized Controlled Trial. Ervhervsstyrelsen will find a company which is suitable for the plugin. They will divide the employees into two groups, one where the plugin works and one where it does not. The authority will then send ”fake” phishing emails to the employees. They will count the number of link clicks of each group. If there is a notable less percentage of link clicks in the group where the plugin does work compared with the other one, the plugin could be beneficial. The test’s purpose is to see if there might be any benefits from using the plugin.

8.1 Improvement of the simulation

The simulation environment can be improved. It is not perfect as many variables were assumed. Further research on individual factors are therefore needed to make the simulation more accurate.

There are multiple different factors that one could improve with the simulation. First off, the results are not conclusive since the test environment was not ideal. More variables would have been beneficial as it could make a more realistic test environment. For example, the information about the distribution of reoccurring emails or the number of emails that people send. Getting more conclusive data for each factor would make for a better simulation and would have given a more accurate result.

References

[1] Anderson, R. John. *Cognitive Psychology and Its Implications*. Worth Publishers, 2009. ISBN: 9781429219488.

[2] Bavel, René van, Vila, Nuria Rodríguez-Priegoa José, and Briggs, Pam. *Using protection motivation theory in the design of nudges to improve online security behavior*. 2019. URL: https://www-sciencedirect-com.focus.lib.kth.se/science/article/pii/S1071581918306475 (visited on 05/06/2019).

[3] Beaver, Kevin. *Hacking For Dummies*. John Wiley Sons, Inc, 2016. ISBN: 9781119154693.

[4] Bhuiyan, M.M. et al. "FeedReflect: A tool for nudging users to assess news credibility on twitter". In: *Proceedings of the ACM Conference on Computer Supported Cooperative Work* (2018), pp. 205–208.

[5] Blumenthal-Barby, J. S. and Burroughs, Hadley. "Seeking Better Health Care Outcomes: The Ethics of Using the "Nudge"". In: *The American Journal of Bioethics* (2012), pp. 1–10. DOI: 10.1080/15265161.2011.634481.

[6] Chouhan, Siddharth Singh. *A Survey of Phishing Attack Techniques*. 2014. URL: https://www.researchgate.net/publication/271156987_A_Survey_of_Phishing_Attack_Techniques (visited on 04/07/2019).

[7] Christian, Reynolds et al. "Review: Consumption-stage food waste reduction interventions – What works and how to design better interventions". In: *Food Policy* (2019), pp. 7–27. DOI: 10.1016/j.foodpol.2019.01.009.

[8] *Classification: True vs. False and Positive vs. Negative*. URL: https://developers.google.com/machine-learning/crash-course/classification/true-false-positive-negative (visited on 05/16/2019).

[9] Company, Houghton Mifflin. *Desensitization*. URL: https://www.dictionary.com/browse/desensitization (visited on 05/06/2019).

[10] *Cybersecurity Challenge*. URL: https://challenges.dk/da/challenge/cybersecurity-challenge (visited on 06/01/2019).

[11] *Danish Business Authority*. URL: https://danishbusinessauthority.dk/ (visited on 06/01/2019).

[12] Dudovskiy, John. *Deductive Approach (Deductive Reasoning)*. URL: https://research-methodology.net/research-methodology/research-approach/deductive-approach-2/ (visited on 04/09/2019).

[13] Dudovskiy, John. *Types of Literature Review*. URL: https://research-methodology.net/research-methodology/types-literature-review/ (visited on 04/09/2019).

[14] Ellenvan, Kleef and Hans C.M.van, Trijp. "Chapter 13 - Methodological Challenges of Research in Nudging". In: *Methods in Consumer Research* (2018), pp. 329–349. DOI: 10.1016/B978-0-08-102089-0.00013-3.

[15] Etemadi, N. "An elementary proof of the strong law of large numbers". In: *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete* (1981), pp. 119–122. DOI: 10.1007/BF01013465.

[16] Fette, Ian, Sadeh, Norman, and Tomasic, Anthony. *Learning to Detect Phishing Emails*. URL: http://reports-archive.adm.cs.cmu.edu/anon/anon/usr0/ftp/isri2006/CMU-ISRI-06-112.pdf (visited on 04/15/2019).

[17] Fienstein, Ken. *How to Do Everything to Fight Spam, Viruses, Pop-Ups, and Spyware*. McGraw-Hill Education, 2004. ISBN: 0072256559.

[18] Graham, Lawton. "Inventing the nudge". In: *New Scientist* (2013), p. 35.

[19] Günther-Hanssen, Christian. *Nudga mot Nätfiske*. 2006. URL: http://cogitocredo.se/nudga-mot-natfiske/?fbclid=IwAR2JGiwzUZ_HQLtId5lgubDbLSQd9DlKXrSo_Kn_2LTRvVKUVXdktsGbf0A (visited on 04/06/2019).

[20] Håkansson, Anne. "Portal of Research Methods and Methodologies for Research Projects and Degree Projects". In: *The 2013 World Congress in Computer Science, Computer Engineering, and Applied Computing* (2013).

[21] Hansen, Pelle. “The Definition of Nudge and Libertarian Paternalism - Does the Hand Fit the Glove?” In: *The European Journal of Risk Regulation* (2016), pp. 155–174.

[22] Huei-Chen, Hsu and Wen-Liang, Liu. “Using Decoy Effects to Influence an Online Brand Choice: The Role of Price–Quality Trade-Offs”. In: *Cyberpsychology, Behavior, and Social Networking* (2011). DOI: 10.1089/cyber.2009.0262.

[23] *Inbyggd elektronik. Övning: Strukturdiagram*. URL: https://www.kth.se/social/files/54cb5575f27654321d77b92b/0_struktur.pdf (visited on 04/15/2019).

[24] Joel, Huber, John, W. Payne, and Christopher, Puto. “Adding Asymmetrically Dominated Alternatives: Violations of Regularity and the Similarity Hypothesis”. In: *Journal of Consumer Research* (1982), pp. 90–98. DOI: 10.1086/208899.

[25] Johnson, E. J. et al. “Beyond nudges: Tools of choice architecture”. In: *Marketing Letters* (2012), pp. 487–504. DOI: 10.1007/s11002-012-9186-1.

[26] Kruck, Gregory P. and Kruck, S.E. *Spoofing – A Look at an Evolving Threat*. 2016. URL: https://www.tandfonline.com/doi/pdf/10.1080/08874417.2006.11645943?needAccess=true@online (visited on 05/06/2019).

[27] Larsen, Henrik, Sørensen, Torben B., and Devantier, Nicolai. *Danskernes informationssikkerhed*. 2018. URL: https://digst.dk/media/18755/danskernes_informationssikkerhed_december_2018_191218.pdf#%5C%5B%5C%7B%5C%22num%5C%22%5C%3A250%5C%2C%5C%22gen%5C%22%5C%3A0%5C%7D%5C%2C%5C%7B%5C%22name%5C%22%5C%3A%5C%22XYZ%5C%22%5C%7D%5C%2C303.307%5C%2C577.1397%5C%2C0%5C%5D (visited on 05/06/2019).

[28] Lehner, Matthias, Mont, Oksana, and Heiskanen, Eva. “Nudging – A promising tool for sustainable consumption behaviour?” In: *Journal of Cleaner Production* (2016), pp. 166–177. DOI: 10.1016/j.jclepro.2015.11.086.

[29] Marshall, Emmanuel. *Indictment: 2016 Democrat campaign hacked using phishing*. 2018. URL: https://www.mailguard.com.au/blog/clinton-campaign-phishing (visited on 05/06/2019).

[30] O'Brien, G. and Davies, M. "Nutrition knowledge and body mass index". In: *Health Education Research* (2007), pp. 571–575.

[31] Pruett, Michelle. *The Average Marketing Email: Open Rate, Click-Through, and More*. 2018. URL: https://www.criteo.com/insights/email-open-rates/ (visited on 05/06/2019).

[32] Ramsberg, Francisca. "När det rätta blir det lätta – en ESO-rapport om ”nudging””. In: *Finansdepartementet* (2016).

[33] Satterfield, Brian. *Ten Spam-Filtering Methods Explained*. 2006. URL: https://www.techsoupcanada.ca/en/learning_center/10_sfm_explained (visited on 04/06/2019).

[34] Scrum.org. *What is Scrum?* URL: https://www.scrum.org/resources/what-is-scrum (visited on 05/03/2019).

[35] Security, Wombat. *State of Phish*. 2018. URL: https://info.wombatsecurity.com/hubfs/2018%20State%20of%20the%20Phish/Wombat-StateofPhish2018.pdf?submissionGuid=2ecea77c-aa0d-404a-b0f4-030732e60a3a (visited on 05/03/2019).

[36] Service, Owain et al. "EAST: Four simple ways to apply behavioural insights". In: *Behavioural Insight Team* (2012). URL: http://www.behaviouralinsights.co.uk/wp-content/uploads/2015/07/BIT-Publication-EAST_FA_WEB.pdf.

[37] Shi, Junxiao and Saleem, Sara. *Phishing*. URL: https://www2.cs.arizona.edu/~collberg/Teaching/466-566/2014/Resources/presentations/2012/topic5-final/report.pdf (visited on 06/01/2019).

[38] Stein, B. S. and Bransford, J. D. "Constraints on effective elaboration: Effects of precision and subject generation". In: *Journal of Verbal Learning and Verbal Behavior* (1979), pp. 329–349.

[39] Symantec. *Internet Security Threat Report*. 2018. URL: https://www.symantec.com/content/dam/symantec/docs/reports/istr-23-2018-en.pdf (visited on 05/03/2019).

[40] Szathmari, Gabor and Kavadias, Nicholas. *Email Impersonation Scams: What You or Your IT Staff Can Do to Protect Your Business*. URL: https://blog.ironbastion.com.au/email-impersonation-scams-phishing-what-your-staff-can-do/ (visited on 05/06/2019).

[41] Thaler, R. H. and Sunstein, C. R. *Nudge*. Penguin Books, 2008.

[42] Zaidi, Shahrukh. *Bypassing Phishing Filters*. 2018. URL: https://delaat.net/rp/2017-2018/p35/report.pdf (visited on 04/06/2019).

Appendices

Appendix - Contents

A First Appendix

The appendix contains the input and output from the simulation, the survey results in Swedish, the scrambler results, the code snippet from simulation and the lists with all the test cases and requirements.

A.1 Input and output

The tested strategy was double. All the input parameters were the same but not the detectionForm for the other strategies. From the following parameters,

```
double startClickPercent = 1.0;
double clickPercent = startClickPercent;
double goodClickPercent = 0.7;
int timesRunning = 1000000;
int numberOfEmails = 1000;
int currentId = 0;
int detectionForm = 0;
double percentBad = 0.0870937;
double percentBadGotLink = 1.0;
double percentGoodGotLink = 0.7;
double percentNewUser = 0.2;
double whiteListCorrect = 0.95;
double sentTo = 0.10;
```

The output was following:

```
run:
averagepF: 0.21347451951443774
spF: 0.0030603015669656316
snT: 0.04122408404829365
timesRunning: 1000000
nudgeTriggerCounter: 118316821
positiveFalseCounter: 101076047
negativeTrueCounter: 16473148
```

```
Total 1000: [positiveFalse%perAllEmail, negativeTrue%perAllEmail
, positiveFalse%pernGoodLinkClick, negativeTrue%perBadLinkClick]
10.1076047
1.6473148
21.347451951443773
48.86155932851716
Total 1000 pF bounds 3.6285476336308074E-7 : [0.21347415665967437
, 0.21347488236920112]
Total 1000 nT bounds 1.831749501464758E-5: [0.488597275790157
, 0.48863391078018625]
Total 1000: [positiveFalse%perAllEmail, negativeTrue%perAllEmail
, positiveFalse%pernGoodLinkClick, negativeTrue%perBadLinkClick]
First 100: [positiveFalse%perAllEmail, negativeTrue%perAllEmail
, positiveFalse%pernGoodLinkClick, negativeTrue%perBadLinkClick]
15.394418
0.62142
32.1201259256689
28.3894075375243
After 100: [positiveFalse%perAllEmail, negativeTrue%perAllEmail
, positiveFalse%pernGoodLinkClick, negativeTrue%perBadLinkClick]
9.520181
1.761303111111111
20.134186254323865
50.28302769290424
First 500: [positiveFalse%perAllEmail, negativeTrue%perAllEmail
, positiveFalse%pernGoodLinkClick, negativeTrue%perBadLinkClick]
10.7036892
1.3411534
22.53078183792637
44.00236332267052
After 500: [positiveFalse%perAllEmail, negativeTrue%perAllEmail
, positiveFalse%pernGoodLinkClick, negativeTrue%perBadLinkClick]
5.284177888888889
```

```
1.0852645555555556
20.156152965674668
52.869926055899846
GoodEmails 9.82759226E8
BadEmails 1.7240774E7
Bad/AllEmails 0.017240774
BUILD SUCCESSFUL (total time: 1 minute 38 seconds)
```

A note, the strings from the output is wrong. Positive false should be False negative and Negative true should be False positive.

A.2 Code Snipplet

The following code is from the simulation part where it is reading and clicking the emails. As said before, this code has been slightly modified from the real one, to make it more understandable. Unnecessary counters have been removed and some comments has been added.

```
for(int i = 0; i < emailBox.size(); i++){
if(emailBox.get(i).gotALink){
if(emailBox.get(i).isDangerous){
medelBad[j]++;
if(Math.random()<clickPercent){
//detection if statement
switch (detectionForm) {
case 0:
//Double
if(!senderWhitelist[emailBox.get(i).idFrom]){
senderWhitelist[emailBox.get(i).idFrom] = true;
}else{
medelnegTru[j]++;
}   break;
//Whitelist
case 1:
```

```
if(!senderWhitelist[emailBox.get(i).idFrom]){
if(whiteListCorrect<Math.random()){
senderWhitelist[emailBox.get(i).idFrom] = true;
}

}else{
medelnegTru[j]++;
}break;
//Sent List
case 2:
if(!senderWhitelist[emailBox.get(i).idFrom]){
}else{
medelnegTru[j]++;
}break;
//Sent list and whitelist
default:
if(!senderWhitelist[emailBox.get(i).idFrom]){
if(whiteListCorrect<Math.random()){
senderWhitelist[emailBox.get(i).idFrom] = true;
}
}else{
medelnegTru[j]++;
}break;
}
}
}else{
medelGood[j]++;
if(Math.random()<goodClickPercent){
//detection if statement
switch (detectionForm) {
case 0:
//Double
if(!senderWhitelist[emailBox.get(i).idFrom]){
```

```
medelposFal[j]++;
senderWhitelist[emailBox.get(i).idFrom] = true;
}   break;
//Whitelist
case 1:
if(!senderWhitelist[emailBox.get(i).idFrom]){
medelposFal[j]++;
if(whiteListCorrect>Math.random()){
senderWhitelist[emailBox.get(i).idFrom] = true;
}
}break;
//Sent List
case 2:
if(!senderWhitelist[emailBox.get(i).idFrom]){
medelposFal[j]++;
}break;
//Sent list and whitelist
default:
if(!senderWhitelist[emailBox.get(i).idFrom]){
medelposFal[j]++;
if(whiteListCorrect>Math.random()){
senderWhitelist[emailBox.get(i).idFrom] = true;
}
}break;
}
}
}
}
}
```

A.3 Original Survey Questions

The Swedish version of the question asked in the survey.

- Vet du vad nätfiske är?
- Har du råkat falla för det?
- Tror du att nudgen skulle kunna påverka ditt beslut?
- Detta är tänkt att används i ett företags sammanhang där det är väldigt viktigt att vara försiktig. Om du såg en sådan varning hur skulle det påverka ditt beteende och i så fall hur?
- Tror du att nudgen skulle vara störande när du öppnade mail? Svara mellan 0-10 där 0 är inte jobbigt alls och 10 är väldigt jobbigt. (Det kommer vara en skala under man kan välja på)
- Hur ofta får nudgen förekomma innan det blir jobbigt?
- Om vi tänker oss tillbaka till ett företag skulle du tycka att Nudgen var jobbig om den kom allt för ofta? I så fall varför och hur skulle man minimera påverkan.
- Hur tyckte ni nudgen såg ut? Finns det något designval ni tänker er ändra/förbättra?
- I ett företag, om denna nudge inte kom upp på ett email skulle ni fortfarande vara kritiska mot emailet eller skulle ni fortsätta att klicka på länken i emailet?
- Om det finns något annat ni tycker är viktigt att nämna skriv det!

A.4 Scrambler Test

Examples of scrambled emails by the scrambler:
from: name.position@domain.com
results:

- name.position59@domain.com
- name@position.com
- position.name@domain.org

- name.position@domain.com
- name.position@domain.org

from: vitek@kth.se
results:

- vitek@kth.com
- vitek@kth.se
- vitek55@kth.org
- vitek2@hotmail.com
- kth@vitek.com

from: taqui@kth.se
results:

- taqui13@kth.com
- taqui@kth.com
- taqui@kth.se
- taqui1@hotmail.org
- kth@taqui.org

from: communication-support@eecs.kth.se
results:

- communication-support.kth@eecs.com
- communication-support4@gmail.org
- eecs@communication-support.org
- communication-support@eecs.kth.se
- communication-support@eecs.com

A.5 List with Requirements and List with Test Cases

Following list A.1 contains the requirements.

Requirement Title	1 - Installation works
Level	0
Description	Must be able to install and have the software program work as intended, which means following our JSP-diagram. The program works as intended if the installation works correctly and all the other requirements are met.
Test cases	• 0.0 - Installation • 0.1 - Complete run 1 • 0.2 - Complete run 2

Requirement Title	2 - Nudge Trigger
Level	1
Description	A dialog triggers and opens when the phishing detection has deemed an email as a possible phishing email.
Test cases	• 1.0 - Nudge Trigger

Requirement Title	3 - Dialog
Level	1
Description	A dialog triggers and opens when the phishing detection has deemed an email as a possible phishing email.
Test cases	• 1.1 - Dialog

Requirement Title	4 - Nudge Trigger - Triggering
Level	2
Description	Be able to create a trigger that can proceed to start functions when a link is clicked. The link can be to a redirecting one or to download a attachment.
Test cases	• 2.0 - Trigger Link • 2.1 - Trigger Attachment

Requirement Title	5 - Nudge Trigger - Initiate Listener
Level	2
Description	Be able to listen on clicks on links or attachments from an email. The listener should only be active in the correct situations, regarding the implementation of how we discover phishing emails.
Test cases	• 2.2 - Event listener

Requirement Title	6 - Dialog - Start
Level	2
Description	To be able to start the dialog on a certain event.
Test cases	• 2.3 - Start Dialog

Requirement Title	7 - Dialog - Proceed
Level	2
Description	The dialog should be able to change it contains, a multiple choice question should be asked and feedback will be returned to the user.
Test cases	• 2.4 - Responsive Dialog • 0.1 - Complete run 1 • 0.2 - Complete run 2

Requirement Title	8 - Dialog - Close
Level	2
Description	The dialog exits correctly.
Test cases	• 0.0 - Installation • 0.1 - Complete run 1 • 2.5 - Exit Dialog

Requirement Title	9 - Email Scrabbler
Level	2
Description	Options of emails can be generated to look alike similarly.
Test cases	• 2.6 - Get Email • 2.7 - Scrabble Emails

Table A.1: List of requirements

Following list A.2 contains the test cases.

Title	0.0 Install
Description	Tests if the program can be installed and ran.
Precondition	The user has access to the code.
Assumption	The user has a browser supported for plugins.
Step	• 1.Npm runs sucessfully • 2. Npm builds without warning. • 3. Upload directory with manifest.json to chrome • 4. Go to gmail on browser • 5. View Console to check if the extension runs accordingly
Expected Results	If the program runs the installation worked.

Title	0.1 Complete run 1
Description	Tests if the program works as expected as the JSP-diagram, testing the normal cases in a pre-made environment that test the basic functionality.
Precondition	The program is running.
Assumption	The program is able to be installed.
Step	• 1. Go to an email client in browser. • 2. Open a mail that is whitelisted. • 3. Click on a link. • 4. Open a mail that is a phishing mail. • 5. Click on a link.
Expected Results	For the whitelisted email the nudge won't appear. For the phishing email the nudge should appear.

Title	0.2 Complete run 2
Description	Tests if the program's scope is correct and doesn't disturb other activities.
Precondition	The program is running.
Assumption	The program can be installed.
Step	• 1. Go to a non-email client. • 2. Click on a link.
Expected Results	The program should not start the nudge dialog.

Title	1.0 Nudge Trigger
Description	Tests if our phishing detection finds emails and listen if a click is clicked.
Precondition	The program is running.
Assumption	A non whitelisted email with a link is ready to be click.
Step	• 1. Open the mail. • 2. Click on the link.
Expected Results	The nudge should be triggers and displayed.

Title	1.1 Dialog
Description	Tests if our nudge is correct regards to planning.
Precondition	Some event opens the dialog.
Assumption	The dialog is ready to be displayed.
Step	• 1. The dialog displays on event. • 2. The user reads the text. • 3. The user clicks to continue. • 4. The user can answer. • 5. The dialog closes.
Expected Results	The pop-up dialog follows the steps correctly with as defined slides.

Title	2.0 - Trigger Link
Description	Tests if a link click triggers an event.
Precondition	A link listener is active, and an email is open.
Assumption	Some basic function ready to be able to activate on trigger.
Step	• 1. Press the link. • 2. See the outcome of basic function.
Expected Results	The outcome for the basic function is correct and is started when event triggers.

Title	2.1 - Trigger attachment
Description	Tests if an attachment click triggers an event.
Precondition	An attachment listener is active, and an email is open.
Assumption	Some basic function ready to be able to activate on trigger.
Step	• 1. Press the attachment. • 2. See the outcome of basic function.
Expected Results	The outcome for the basic function is correct and is started when event triggers.

Title	2.2 - Event listener
Description	Tests if a listener initiates correctly regards to our implementation criteria.
Precondition	The plugin is ready and active.
Assumption	A ready email that fits the criteria to create listener.
Step	• 1. Open the ready email. • 2. Look if the event listener was created. • 3. Go away from the email. Unfocus it.
Expected Results	The listener should be active, while the email is in focus.

Title	2.3 - Start Dialog
Description	Tests if the dialog which will include the nudge can be open by an arbitrary trigger.
Precondition	A dialog ready to be open.
Assumption	An event that calls the start function.
Step	• 1. Trigger the function that opens the dialog. • 2. Wait until the dialog displays.
Expected Results	The dialog should open the function is called.

Title	2.4 - Responsive Dialog
Description	Tests if the dialog can respond to input and behaves correctly regards to the input.
Precondition	A dialog that is open.
Assumption	The opened dialog can handle inputs.
Step	• 1. Use the inputs options of the dialog. • 2. Note what is happening.
Expected Results	The inputs should have correct outputs by the dialog.

Title	2.5 - Exit Dialog
Description	Tests if the dialog can be closed after the dialog is done.
Precondition	A dialog that is open.
Assumption	-
Step	• 1. The dialog doesn't close when nothing is input. • 2. After the correct command it exits.
Expected Results	The dialog should be closed.

Title	2.6 - Get Email
Description	Tests if able to fetch the senders real email address.
Precondition	An email ready.
Assumption	A program ready to fetch.
Step	• 1. Try to fetch the email address • 2. See if the correct email address is fetched.
Expected Results	The correct email address can be seen.

Title	2.7 - Scrabble Emails
Description	Tests if from a given email, generate emails of the same kind.
Precondition	A ready email address.
Assumption	The email can be scrabbled.
Step	• 1. Activate the scrabbler. • 2. See if the emails results look alike.
Expected Results	If the test has the same output as the model of the scrabbler has.

Table A.2: List of test cases

TRITA-EECS-EX-2019:585