Bachelor Degree Project

Masquerader Detection via 2fa Honeytokens

Author: Anton Wiklund
Supervisor: Daniel Toll
Semester. VT 2021
Subject: Computer Science

Abstract

Detection of insider threats is vital within cybersecurity. Techniques for detection include honeytokens, which most often are resources that, through deception, seek to expose intruders. One kind of insider that is detectable via honeytokens is the masquerader. This project proposes implementing a masquerader detection technique where honeytokens are placed within users' filesystems in such a way that they also provide Two Factor Authentication(2fa) functionality. If a user's second factor – the honeytoken – is not accessed within a specified timeframe after login, this indicates a potential intrusion, and only a "fake" filesystem will remain available. An alert is also triggered. The intention is to deter insiders from masquerading since they are aware that they must access a uniquely located honeytoken after logging in to the legitimate user's account. The technique was evaluated via user-testing that included interviews, a checklist with requirements for feasibility, and a cyber-security expert's opinion on the technique's feasibility. The main question evaluated during the project was the feasibility of adding the proposed technique to a computer system's protective capabilities. The results of the project indicated that the proposed technique is feasible. The project's results were also compared with the results of prior related research. The project's scope was limited to a Linux system accessed via SSH into a Bash terminal(non-GUI-compatible), and the implemented technique was also evaluated within such an environment.

Keywords: *Cyber security deception, two-factor authentication(2fa), honeytoken, misinformation, intrusion detection.*

Preface

I dedicate this Bachelor's degree project to my Grandfathers (Sven Erik Nordgren & Karl-Erik Wiklund) and my Grandmothers (Ann-Kristin Nordgren & Sanna Wiklund). Especially I thank Sven.

Immortality

Do not stand
By my grave, and weep.
I am not there,
I do not sleep—
I am the thousand winds that blow
I am the diamond glints in snow
I am the sunlight on ripened grain,
I am the gentle, autumn rain.
As you awake with morning's hush,
I am the swift, up-flinging rush
Of quiet birds in circling flight,
I am the day transcending night.
Do not stand
By my grave, and cry—
I am not there,
I did not die.

- Clare Harner, 1934

An extra thanks to my supervisor Daniel Toll of Linnaeus University, who helped me a lot during the research and writing of this project.

Contents

1– Introduction

This project is a 15 HEC Bachelor's thesis in computer science, focusing on cybersecurity. The specific area of cybersecurity focus is intrusion detection, which spans a wide array of techniques and use-cases. The core objective of intrusion detection is to detect intruders(i.e., unauthorized users).

Intrusion detection is an important aspect of cybersecurity since an unnoticed intruder can access information and services only intended for authorized users. In a worst-case scenario, an intruder can alter the core of the compromised system, overriding firewall rules, implementing backdoors, and taking over the whole system. In severe cases – such as the 2020 Solarwinds compromise [1] – a breached company's developed source code may become altered and shipped to other companies, which also become compromised.

The targeted readers for this project are those with a somewhat limited understanding of the IDS and masquerader concept but who would like to expand their knowledge. However, a basic understanding of different kinds of IDSs and how they seek to expose intruders is expected. Since no research on the 2fa honeytoken approach this paper proposes was found, we hope those with cybersecurity experience would find this project to be of at least some interest.

This project's societal and industrial motivation was to provide an additional technique for validating that a user is who it purports to be. Many programs with this kind of functionality are also proprietary, so an additional free and open source variant might be relevant for companies with smaller budgets. According to the Swedish intelligence establishment, cyber-related industrial espionage has also intensified during recent years, leading to significant losses of protected knowledge(which affects society as a whole) [2]. The scientific motivation was to explore the potential value of a 2fa technique where the 2fa key is located within the filesystem while also allowing for intruder detection. Papaspirou et al. provided an outline of currently available 2fa techniques but did not mention the technique proposed in this project [3]. Due to being unable to find a similar technique during the background research, the conclusion drawn was that a valid scientific motivation did exist.

The project itself was a feasibility study. The research methodology Design Science was implemented iteratively to develop and evaluate a 2fa honeytoken prototype. The prototype was used to gather results to answer research questions with. After results had been gathered, their reliability and

validity were analyzed and the research questions themselves answered.

1.1 – Background

One of the most challenging cybersecurity threats today is, according to Homoliak et al., insider threats. At the same time, these threats are not well handled by conventional cybersecurity solutions [4]. The effect of this can be related to how, according to Peng & Wang, IP theft affecting American companies in around 60% of the cases can be traced back to insider activity [5]. According to a survey conducted by Vormetric Inc, 89% of companies consider themselves vulnerable to hostile insider activity [6]. The fact that so many companies consider themselves vulnerable to insider activity implies that solutions are needed.

In cybersecurity terminology, one kind of insider is called the "masquerader" [7]. A masquerader is an unauthorized entity who has gained access to the credentials of an authorized user and then uses these credentials for various kinds of activity – harmful or not [4]. A masquerader can be both an insider or an external entity [4].

Two relatively common intrusion detection techniques are honeypots and honeytokens. Both of these are deception oriented techniques(see *Chapter 3.4)*, and the main feature which distinguishes them from one another according to Spitzner – whose definition is seemingly still considered the unofficial truth – is that a honeypot is a complete computer system which is populated with fake information. In comparison, the honeytoken can be (1)a deceptive disinformation file within an otherwise real system or (2)a fake login service. In general, a honeytoken/pot represents a sort of "trap" intended to expose an intruder and analyze its intentions and methods.

2fa is a technique where two authorization techniques are integrated to provide more robust access-control security. For example, the user enters the password, and then a code is sent to the phone number associated with the account. The user also enters the received code (the user must be authorized via two factors: the password and the phone number). One core challenge of this project was to implement the 2fa honeytoken so that it would also provide a means for exposing a masquerader. The proposed technique did not have to be *better* than existing intrusion detection techniques but rather a *relevant addition* to existing techniques.

1.2 – Related work

The three articles that this project mainly built upon are described in this section to establish this project's context. It is also stated how they relate to – and differ from – this project's proposed technique. Hopefully, doing so will provide the reader with a good overview of how prior research has influenced the project's proposed technique. For more details on the concepts that the proposed technique implemented, see *Chapter 3*.

Stolfo et al. outlined a technique for preventing the theft of valuable data by masqueraders, by monitoring user behavior. They proposed the use of decoy documents spread within the system. If these documents were used in some way, it would indicate suspicious behavior [8]. Their paper was relevant since this project proposes a technique for exposing user behavior that might indicate a masquerader. While the decoy documents proposed by Stolfo et al. needed the intruder to actively use them in some way to alert the admin, the monitoring proposed by this project's 2fa honeytoken depended upon the intruder *not* using it – if the 2fa token was not used, this implied suspicious behavior.

Papaspirou et al. outlined an overview of the strengths and weaknesses of currently available 2fa methods and how 2fa honeywords(fake passwords, indicating suspicious behavior if used) can be implemented to expose intruders [3]. Their paper was relevant since this project's proposed technique integrates 2fa functionality and intrusion detection. They did not include a 2fa solution where the 2fa token exists within the filesystem and is accessed after login, which was this project's proposed 2fa solution.

Ahmad et al. outlined various ways misinformation can be presented and transmitted. They also contextualized the historical use of cyber misinformation, and discussed its potential value [9]. Their paper was mainly relevant because it provided an overview of how the proposed technique could implement misinformation with honeytokens. Ahmad et al. did not integrate misinformation presentation with 2fa functionality, which this project did.

The three papers mentioned above all represent – to various degrees – specific aspects of this paper's proposed technique. They divert from the proposed technique in two main ways: neither integrated all these various aspects(2fa, honeytokens, misinformation) into a single technique, and nor did any of them discuss a technique where honeytokens located within a filesystem are used to prevent unauthorized access by intruders and alert admins on suspicious behavior.

Finally, a technique such as the one proposed in this paper was not

found during background research, concluding that the proposed technique probably did fill a knowledge gap by exploring the concept's feasibility. For example, Papaspirou et al. discussed various available 2fa techniques, but did not mention the proposed technique [3]. Grimes also outlined different ways in which 2fa can be implemented, and stated that one method is to give the user something that only it knows – mentioning passwords, patterns to be drawn, and PIN-codes. Notably, he did not include a 2fa token hidden within the system, with only admins and the authorized user supposedly knowing its location [10].

1.3 – Problem formulation

In this section, a brief description of the general problem will first be presented. Following this, descriptions of the two research questions will be broken down into sub-questions and presented.

The detection of intruders is vital when it comes to the protection of a computer system. If intruders are not detected, the whole company or organization may become vulnerable. One means of defense is the implementation of user access-control. The intention behind this project's proposed technique was to provide an additional means for solving the problem that intruders often bypass the access-control mechanism by masquerading as an authorized user, thereby gaining access to, for example, the user's files. This technique was designed as a 2fa honeytoken and intended to solve the core challenge stated in the final paragraph of *Chapter 1.1 (*to develop it to provide masquerader detection). The overall goal of the project was to explore the feasibility of the proposed 2fa honeytoken technique, and how it compared to prior techniques. To do so, the goal was reformulated as two research questions.

Research Question 1: *is the technique feasible?* The first research question concerned the proposed technique's general feasibility (simply if it might constitute a relevant and functional protective mechanism). This question was broken down into three subquestions: Could users interact with the technique successfully(1.1)? Could the technique fulfill a large number of feasibility-related aspects(1.2)? Would a cyber-security expert consider the technique feasible(1.3)?

Research Question 2: *how does the proposed technique compare to prior 2fa and honeytoken implementations*? The second research question concerned how the technique proposed in this paper compared to existing 2fa and honeytoken techniques. It is important to compare a new and early-stage

technique to similar but more established techniques (a selection of which were presented in *Chapter 1.2)*. This question was broken down into three subquestions: What were the advantages/disadvantages of this new technique compared to prior techniques(2.1)? Which use-cases might the technique be better/worse suited for(2.2)? Can the results of prior research be used to argue that this project's proposed technique is relevant(2.3)?

1.4 – Outline

Chapter 1 has outlined the general premises of this project. *Chapter 2* will present a more detailed discussion regarding the methodologies and research questions employed to fulfill the project's goals and answer the research questions. *Chapter 3* provides the details regarding the prior research upon which this project was built. *Chapter 4* provides an overview of the implementation of the proposed technique, including an algorithm and a diagram. In *Chapter 5*, the reader is presented with results gathered during the technique's development. These results will also be analyzed. *Chapter 6* discusses the project as a whole, particularly regarding the reliability and validity of the results used to answer the research questions. *Chapter 7* includes the overall conclusions of the complete project, most notably in regards to the final answers to the research questions and whether the proposed technique should be considered feasible or not. The project's potential contributions are also discussed here. *Chapter 8* is the final chapter and is where future research is discussed more in-depth than in preceding chapters.

2 – Method

In this chapter, an overview of the methodologies that were adopted for the project will be presented. By reading this chapter, the reader will hopefully understand the underlying processes that guided the project. First, the research questions will be raised and mapped to the methodologies employed for answering them. Before these methods themselves are presented, however, the design science(DS) research pattern and how it was used during this project's iterative development of the 2fa honeytoken prototype will be explained. The reader will thus be provided with an overview of how the actual research process was structured. After this, the methodologies used during individual iterations, and employed to answer the research questions, will be described in detail. A plan for improving the validity and reliability of the results will also be presented. Finally, the ethical considerations of the project, and the justification for choosing the feasibility methodology will be presented.

First of all, however: as stated, this project is a feasibility study. A feasibility study aims to determine whether an idea is generally functional within a specified context. A final evaluation determines whether the idea is feasible and should be researched further. A feasibility study is a preliminary result for determining the potential of an idea [11]. Since the intention of this project was to determine the potential relevance of the proposed 2fa honeytoken technique, a feasibility study seemed to be the most reasonable choice.

2.1 – Research Questions

In this section, the research questions will be described in more detail. The individual research questions will also be mapped to the specific methods used to provide answers to them. These methods are presented in *Chapter 2.4*.

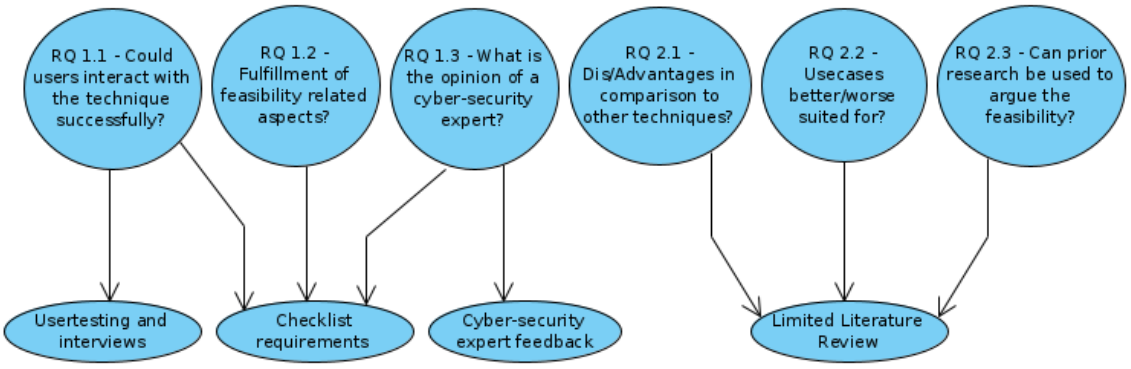


Diagram 2.1 – Methods used for gathering information on and solving individual research questions

The first RQ, concerning the general feasibility of the technique, was evaluated by breaking down the research question into the three more specific questions specified in *Chapter 1.3* and *Diagram 2.1.* RQ 1.1 was answered by setting up a server where users could test the technique and be live interviewed. RQ 1.2 was answered via a checklist with various requirements for the implemented technique to be considered feasible. RQ 1.3 was answered by forwarding the tool and the report to a cyber-sec expert – Ola Flygt of Linnaeus University. This third sub-question was ascribed critical importance in regards to the feasibility of the technique. According to Castro & Mylopoulos [11], after a feasibility study, management decides whether the proposal is feasible or non-feasible and makes a go or no-go decision on further development/research. Management was, in this case, represented by the cyber-security expert. The methods for RQ 1.1 & 1.2 & 1.3 are described in *Chapter 2.4.*

Question two, regarding how the proposed technique compared to prior techniques, was also broken down into three separate sub-questions, which were also specified in *Chapter 1.3* and *Diagram 2.1.* Sub-question 2.1 was answered by comparing the various results of this project with the results of prior peer-reviewed projects. Sub-question 2.2 was answered by drawing general conclusions on how similar techniques are seemingly used. Sub-question 2.3 was answered by considering suggestions on future research mentioned by authors of peer-reviewed papers and then comparing them to the results of this project. The main method employed for answering the sub-questions of question two was a limited literature review (see *Chapter 2.4.4)*.

2.2 – Design Science

In this section, the reader will be provided with an overview of the DS methodology [12]. Details on how the DS stages were incorporated into the project will be presented in *Chapter 2.3*.

First, DS is the attempt on the part of Peffers et al. to provide a "formally defined" methodology for conducting Information Systems research and is structured according to the following six stages [12]:

1. *Identify Problem and Motivate.* Includes, for example, the problem formulation (see *Chapter 1.3*). This stage presents researchers with the opportunity to communicate what the challenge of the project was and why researching the challenge was of value.
2. *Define Objectives of Solution.* For example, concerns creating research questions and mapping these research questions to specific research methods (see *Chapter 2.1)*. The intention is to provide a means for solving the problem formulation.
3. *Design and Development.* During this stage, the researcher designs and creates the artifact supposed to help answer the research questions. According to Pefferts et al., this includes "determining the artifact's desired functionality and its architecture and then creating the actual artifact" [12].
4. *Demonstration.* It can be a single act of demonstration to show that the idea behind the artifact can be implemented. This was represented by the final version of the tool functioning according to the functional requirements specified (see *Chapter 5.2)*.
5. *Evaluation.* The current results are now evaluated to, for example, determine which research questions have been answered. It concerns "comparing the objectives of a solution to actual observed results from use of the artifact in the demonstration" [13]. The evaluation itself can "take many forms" [13] and depends on the research methodologies employed.
6. *Communication.* Now the project and its results are communicated to other people. Communication can occur, for example, by publishing the project paper in a journal. Communication generally leads to feedback and potential improvement of the research. The final results of this project were communicated to a cyber-security expert (see *Chapter 5.3)*, and the paper itself was published.

The nominal DS process was employed during the project since this is the

"problem-centered approach," and the core of this project is the problem formulation of *Chapter 1.3*. The nominal process includes all of the six different DS stages. It is, however, worth noting that Peffers et al. stated that "there is no expectation that researchers would always proceed in sequential order from activity one through activity six" [14]. This flexibility was taken into account during the "Develop Prototype Further" phase of the iterative process – when it was sometimes necessary to return to the literature research phase (see *Diagram 2.3*).

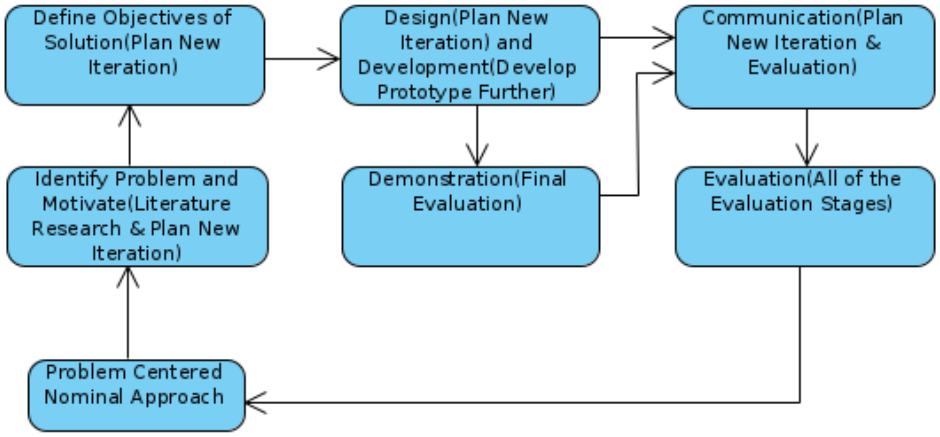


Diagram 2.2 – Diagram of how the DS stages were mapped to the iterative process of Chapter 2.3

2.3 – The Iterative Development Process

In this section, the details on how the iterative development process was structured will be presented. This process was built on the DS methodology presented in *Chapter 2.2* and is visualized in *Diagram 2.3 (*see *Diagram 2.2* for additional context). In this diagram, the iterative process and the stages that it included are presented.

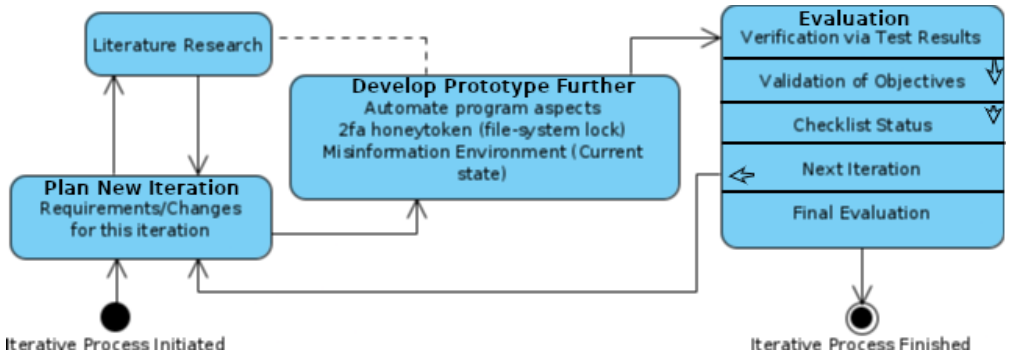


Diagram 2.3 – The development of the technique was structured according to the diagram's iterative development process

In the below bullet list, each phase of the iterated process presented in *Diagram 2.2* is defined. In the parenthesis, they are mapped to their respective DS stage.

1. *Plan New Iteration (Identify Problem and Motivate & Define Objectives of Solution & Design)* – The first step in the iterative process was to determine the requirements to be implemented for that iteration and their motivations.
2. *Literature Research (Identify Problem and Motivate)* – Step two and step one interacted in a sort of feedback loop, where decisions for the new iteration would be compared with the literature to enable better decisions to be made.
3. *Develop Prototype Further (Development)* – During step three, the tool was developed to a state where it would fulfill the iteration's requirements. During this stage, it was sometimes necessary to return to stages 1 and 2 to research/plan a better way of developing the technique.
4. *Evaluation (Demonstration & Evaluation & Communication)* – During the final step of the iterations, the results of the current iteration were evaluated. Only if the core requirements had been fulfilled adequately would the iterative process finish. The final part of the evaluation of each iteration was to determine what parts of the tool should be kept as they were and which should be further developed. After this, the next iteration was initiated.
5. *Final Evaluation (Evaluation & Communication)* – During the end of the final evaluation, the tool was presented to a cyber-security expert,

and analysis regarding how to make the tool relevant for real-world usage was subsequently made(see *Chapter 5.3)*. The paper was also published.

The various steps of the DS methodology were also revisited during individual iterations. For example, if a problem occurred during *Develop Prototype Further (Development)*, then instead of researching it during the next iteration, the process was backtracked. First to *Literature Research (Identify Problem)* and then *Plan New Iteration (Design)* so that the problem could be researched, planned, and a solution then implemented during the current iteration.

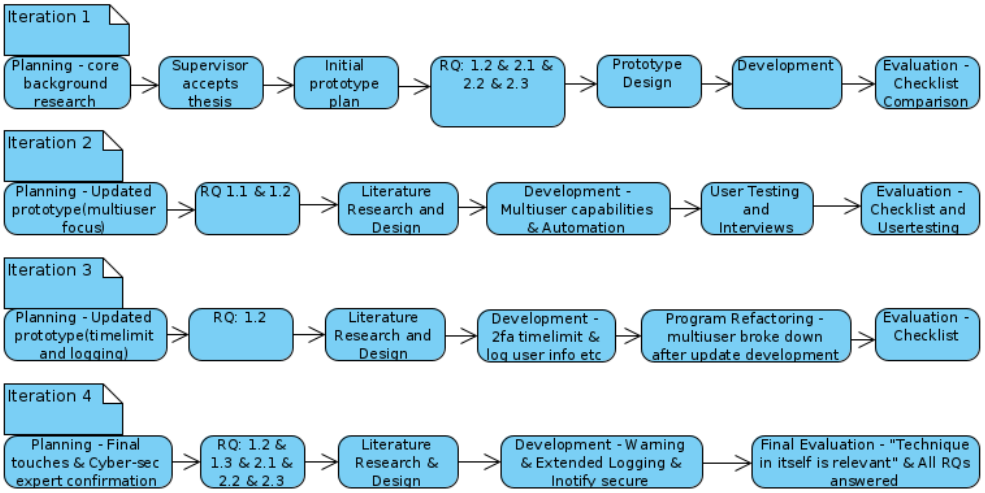


Diagram 2.4 - describes the process for each individual iteration. This includes aspects of the planning and the RQ concerned. It also includes details on functionality developed during the iteration, as well as the testing and evaluation of the technique.

The first iteration was planned from the start of the degree project until the first meeting with the supervisor. The core effort of the first iteration was the background research (see *Chapter 3)* and the planning of how the project research should be structured (see the complete *Chapter 2)*. However, it also included the design of the basic premises of the technique and tool – including the development of an initial prototype.

The second iteration was the iteration when user-testing was employed during the evaluation (see *Chapter 2.4.1)*. It was therefore decided that the second iteration would add functionality for multiple users being tracked and honeytokens being created automatically from within the tool.

The main focus of the third iteration was to implement a fully

functional time limit. This time limit would protect the 2fa honeytoken from being accessed after a specified number of seconds. At the same time, also the general misinformation environment would be made more realistic.

The fourth iteration was planned as representing the final iteration. Due to this, all aspects not scratched from the checklist (see *Chapter 2.4.2)* during previous iterations were added as goals for this final iteration. The most important aspect of the final iteration was the feedback received from the cyber-security expert (see *Chapter 2.4.3)*.

2.4 – Research methods

In this section, an overview of the research methods employed during the project – to answer the research questions and provide a solution to the problem formulation – will be presented and motivated. The research questions were mapped to the methods that will be presented and motivated in this section. The results of all sub-methods are presented in *Chapter 5*.

First of all, since DS is a multi-method research pattern, this allowed for flexibility regarding research methods that could be employed. The methods themselves are described further down, but what they all had in common was that they were oriented towards qualitative instead of quantitative research [14].

2.4.1 – Usertesting and Interviews, RQ 1.1

To determine if users could successfully interact with the technique, interviews with testers – four of them, all computer-science students in their third year – were conducted while testing the technique. The interviews were both structured and unstructured – meaning that both predetermined questions existed, while they could also speak freely about their opinion on the tool. The user-testing was conducted during the second iteration of development on a Ubuntu server. Testers would SSH into this server. The tool ran from the root account on the server while the testers had a normal user account without sudo functionality. During testing, the users were informed about various aspects of how the tool functioned. It was also demonstrated how their interactions with the 2fa honeytoken affected their access to the hidden folder. During the testing, the interview process was unstructured to provide as much freedom as possible for the testers to give feedback on the technique. After testing was completed, the interview process was structured to get answers to the most important questions.

The testers' answers were analyzed not only individually but also in relation to one another (to help determine the degree to which their opinions

on the technique‘s feasibility should be considered relevant). See *Chapter 5.1.1* for this analysis.

The below questions were chosen for the structured part of the interviews. The motivation for them is also stated here.

1. *Do you think this might be a relevant technique to develop further*?
   Since evaluating the technique’s feasibility was the core objective of this project, it was important to determine if the testers considered the technique relevant.
2. *Was the 2fa honeytoken irritating to use?*
   Since how new users experienced the 2fa honeytoken was of great relevance in regards to how likely it was that they would use the token correctly, affecting the feasibility of the proposed technique.
3. *Potential weaknesses of the technique?*
   This question was asked to explore whether or not testers could provide relevant suggestions for further developing the prototype. If they would be able to do so, it might indicate that they had a good understanding of the technique. If they had a good understanding of the technique, their opinions on the feasibility would be more relevant.
4. *Where would you hide the 2fa honeytoken?*
   To get feedback for potential ways to implement the 2fa honeytoken better and to determine the testers’ understanding of what the proposed technique was intended to achieve.
5. *Potential strengths of the 2fa honeytoken?*
   To see if the testers would acknowledge the core functionality as a potential strength without being explicitly told that this was an intended strength.
6. *How would you design the honeytoken?*
   The main intention for this question was to receive feedback that would allow the tool to be developed in a more relevant fashion for users.

The interview results were then translated from Swedish into English and edited to remove unnecessary words and improve grammar (without this affecting the important aspects of the statements). The answers were then used for answering RQ 1.1. The results of the user-testing and interviews are found in *Chapter 5.1*. Only the most important details of the tester’s answers are found there, however. For the complete translated interview answers, see *A. Appendix*.

In summary: the testers were four in total, and all of them were computer science students in their third year - three males and one female. They were contacted via Slack, and the interview process also occurred via a Slack video call (30-60 minutes per session depending on the tester's interest). Participants were using their own computers, and since all of the tools ran on the server details on their computers were not gathered. The server that ran the tool itself was a Ubuntu Server 18.04 with a single 2ghz CPU-core and 4GB RAM. The testing process occurred at each individual tester's home, and was not parallel. Only one tester tested the tool and was interviewed at any single time. The interviews were transcribed in real-time to a word document, and stored in a regular folder as anonymized documents (each document was named after the initial letter in the tester's first name). Testers gave permission for their information being stored in this way. None of the testers thought that the information was important to protect in an extensive manner. For details on the information that the testers were provided with, see *B. Appendix.*

2.4.2 – Implementation and Requirements Checklist, RQ 1.2

During this method, an implementation of the proposed technique was created (see *Chapter 4*). To be able to determine whether this implementation proved feasibility or not, a requirements checklist was created. After the points were fulfilled, the results were analyzed to determine if they strengthened the case for feasibility. Finally, the reliability and validity of the checklist results were considered and are discussed in *Chapter 6*.

The below checklist represents the requirements that the final artifact would need to have fulfilled. The section to the left is the requirement, while the right is a motivation for why it was important to implement this aspect.

1. Always responds to 2fa activity.	If the tool could not respond to the 2fa honeytoken activity in a 100% reliable way, then the technique was not reliable. That would undermine feasibility.
2. Always locks/opens folder.	If the tool could not open/close the folder in a 100% reliable way, then the technique was not reliable. That

	would undermine feasibility.
3. Time limitation is functional.	Without a time limitation for accessing the 2fa token after login, it would be too vulnerable for scans.
4. Testers can use program easily.	Security had to be user-friendly, so that users did not skip important steps or make mistakes that an intruder could exploit.
5. Tool is not resource intensive.*	If the tool used too much resources, then it might affect computer systems in a negative way.
6. cyber-security expert says idea is feasible.	Very important, since an expert has a good understanding of what is needed in the field.
7. Logs user info when 2fa not accessed within timelimit.	If a user logs in and does not use the 2fa token it might indicate an intruder. Feasibility of the tool would therefore be strengthened if such occurrences were logged.
8. 2fa honeytokens and users can be added automatically.	Modern programs generally automate simple tasks. A list of users and locations should for example be possible to add, so that the tool could create all of the relevant tokens and folders.

9. Warning is sent to admin if 2fa honeytoken is not accessed within the timelimit?	Builds on point 7. If a potential intruder were active, admins must be notified quickly so that they can take preventive measures.
10. Tool can run with several users active at the same time.	If the tool could only run for one user, then the tool could only protect a very limited number of systems that are in need of protection.
11. Does not need to be restarted.	If the tool requireed active intervention after user interactions, then it would introduce too much unreliability into the system.
12. Researched how Inotify can be circumvented, and implemented potentially necessary protection against this.	So that users and intruders were unable to gain information about the locations of the 2fa honeytokens. If they were able to gather this information, the tool would serve no purpose.
13. Remove commandline history, so that a masquerader can't find the 2fa honeytoken in the .bash_history.	Builds on point 12. All information about 2fa honeytoken locations must be hidden from everyone except admins and the individual users.

** The tool not being resource-intensive was defined as: the tool could run on an eight GB, 2.6ghz 2core CPU laptop, while several users logged in and out, without this affecting the laptop's general usability in any way.*

The details on the implementation itself are found in *Chapter 4*. There, the reader will find a pseudo algorithm for the implementation and a diagram visualizing this algorithm. The results of the implementation

requirements checklist provided data for answering RQ 1.2 and are found in *Chapter 5.2.*

2.4.3 – Cyber-security Expert Feedback, RQ 1.3

When the technique fulfilled all of the checklist requirements, the source code, this report, a video showcasing the tool, and the source code itself, was sent to the cyber-security expert. Two questions were asked in relation to this: Is this technique feasible? What should future research focus on? By doing this, it would be possible to argue that the technique is feasible not only according to a student-created checklist and user-testing only involving students but also according to someone who is a recognized expert within the field (representing the manager's go-ahead [11]).

The cyber-security expert's feedback answered RQ 1.3 and point number six on the checklist (related to RQ 1.2). The results of the cyber-security expert's feedback are found in *Chapter 5.3.*

2.4.4 – Limited Literature Review, RQ 2

During the limited literature review, the three papers specified in *Chapter 1.2* were chosen in order to provide a specific context within which research question two could be answered. In addition to analyzing the three papers referenced, a more general approach was also taken. This approach included identifying suggestions on future research that occurred in a large number of papers (this mainly concerned the integration of individual techniques with one another and the value of cyber-security deception. See *Chapter 3.4 & Chapter 3.8).*

The limited literature review was conducted simply by using google scholar, and then accessing the articles via Linnaeus University's library. By searching for keywords such as "2fa honeytoken" & "cyber security deception" & "honeytoken misinformation", it was possible to find articles relevant to the various subjects of this project (for complete search queries, see *C. Appendix)*. Articles were analyzed and sorted after apparent relevance, and when "enough" of them had been gathered, they were used in order to establish a foundation for the project (see *Chapter 3)*. The background theory was also used to communicate the current status of research and relevant concepts that the proposed technique was built upon. The main reason for completing the limited literature review was to gather the information necessary to determine the answers to RQ 2.1 & 2.2 & 2.3. Without a solid understanding of the theory, it would be impossible to answer these questions and place the project within a relevant cyber-security context. In addition to

this, the literature review was conducted to understand better how various aspects of the proposed technique could be designed and implemented. The results of the limited literature review are mainly found in *Chapters 3 & 6.3*.

2.5 – Reliability and Validity

In this section, the reader will be presented with how the aspects of validity and reliability were handled. This will describe how the author tried to improve the relevance of the project results. First, the overall methods for evaluating reliability and validity will be presented. After this, the plan for increasing the reliability and validity of the results will be presented.

During the analysis of the project's results, general "observations and rules of thumb" were taken into consideration since, according to Brooks (1988), this represents data that can be valuable as preliminary results [15]. A feasibility study is a form of preliminary result (see intro to *Chapter 2)*. The gathered results were mainly subjective and therefore needed to be analyzed from a point-of-view that took more general "observations and rules of thumb" into consideration. Also, M. Shaw stated that early reports within a specific area can be less formal and focus more on qualitative aspects while stating a *Persuasive* validating case [13]. Since the interviews and the checklist would not represent strictly objective data, the analysis of why their results are reliable and valid therefore focused on persuasive arguments.

In addition to the above, the validation of the results relied upon the aspect of "experience." The tool was used in an actual system environment, not only by the author but also by other people. This is, according to M. Shaw, an aspect of the qualitative model and a valid means for gathering results. M. Shaw, for example, stated that "if the original question was about feasibility, a working system, even without analysis, can be persuasive" [13]. A working implementation was created (see *Chapter 4*).

2.5.1 – Validity

The concept of validity [13] relates to analyzing results and drawing conclusions from these results in a way that is relevant. The relationship between the results themselves and what they are argued to represent must be logical. Conclusions on the validity of a project's results must be limited to the context of the project itself. For example: since the testers of the tool are computer science students, then the validity of their opinions is not enough to argue that the proposed technique is feasible for a real-world context. Close attention was also diverted to the C*onstruct Validity*. For example, the

questions in the interviews with testers were written so that they were closely related to the research questions presented in *Chapter 2.4.1* [14].

2.5.2 – Reliability

Reliability concerns the extent to which results represent a correct picture and if the results can be replicated. For example, problems in relation to reliability can occur if incorrect methods for analysis and data gathering are used [13] or if external factors affect the testing process. The reliability was evaluated concerning experience (user-testing), which focuses on the technique being "used on real examples by someone other than me" [14]. The reliability was also evaluated against the checklist. For the checklist, the intention was to add aspects closely related to feasibility, which also could be used for future evaluation of the technique. A checklist focused on aspects of feasibility should strengthen the reliability of the project since it could be used in order to try and replicate it. In particular, one point was added to strengthen the reliability of the checklist: the cyber-security expert's opinion on the feasibility.

2.5.3 – The Plan for Improving Validity and Reliability of Results

In an attempt to limit the risk when evaluating the reliability and validity of the user-testing results (see *Chapter 5.1*), the following aspects were considered:

- *Were the interview questions of good quality?*
- *What knowledge of cyber-security did the testers have?*
- *Did the testers exhibit a good understanding of the technique?*
- *How extensive engagement did the testers exhibit?*

When it came to the results of the checklist (see *Chapter 5.2)*, their reliability and validity would be considered in relation to the following aspects:

- *Were enough aspects related to feasibility included?*
- *Did the result of each aspect strengthen feasibility?*
- *Were the aspects, in general, reasonably complex?*

When it came to the results of the literature review (see mainly *Chapter 6.3)*, their reliability and validity would be considered in relation to the following aspects:

- *Was it possible to find peer-reviewed articles highly relevant to the proposed technique?*
- *Did the supervisor and cyber-security expert consider the background research to have been extensive enough?*

The manager's go-ahead (discussed in *Chapter 2.4.3* and with results in *Chapter 5.3)* was determined to be vital concerning the reliability and validity of the project as a whole. Only if the cyber-security expert gave this go-ahead would the technique's feasibility (RQ 1) be considered successfully validated. Requiring a third party to evaluate the tool would also strengthen reliability since someone recreating the tool could contact the stated cyber-security expert to see if he also considered the new implementation feasible.

For the discussion on the reliability and validity of the answers to the research questions, see *Chapters 6.1 & 6.2*.

2.6 – Ethical Considerations

In this chapter, a summary of ethical considerations will be presented. Since the testing and development of the project only involved the author of the paper, the testers, and the cyber-security expert, the project was isolated from having potential real-world consequences. Due to this isolation, the ethical considerations were exclusively related to how testers should be informed about the testing process and how the information they provided in relation to testing would be stored. The following ethical considerations were identified:

- Since the testers would be active on the same server, they would have to consent to this.
- Since the behavior of testers would be monitored, it would be necessary to inform them of how this would occur so that they could consent.
- Testers' personal data would be handled in an anonymized fashion. Testers were so few that it was unnecessary to write their names down anywhere. Instead, they could be identified as a single letter.

The subject of ethics and legal circumstances were discussed by Fraunholz et al. [17]. They provided an overview of several articles dedicated to the subject, and their discussion was considered concerning this project's ethics. A short document was also written, which provided the testers with a good understanding of the tests they would participate in. This document included relevant information about how testing would occur and how the testers' data would be saved. See *B. Appendix* for the document testers were provided.

Before the testing, all of the testers agreed to the premises. None of the testers objected to any of the aspects of the user-testing. They were ok with sharing the server with the other testers, having their interviews stored anonymized, and their testing being live monitored.

2.7 – Excluded Research Methodologies

In summary, this project intended to determine whether or not the proposed technique was workable, and a feasibility study is the most common methodology for exploring this. Research methodologies that, however, were excluded due to them being less suitable included:

1. *Systematic literature review* – this was not possible to do since the proposed technique did not already exist. Therefore, there did not exist previous research that could be used to explore the relevance of this project's proposed technique.
2. *User survey* – this methodology is seemingly relevant when it comes to a large number of people who have already interacted with the technique for some time. For this project, it was assumed that greater value was found in the detailed response from a limited number of testers than in the relatively static survey responses of many testers.

Methodologies for validation and analysis not used included :

1. *Systematic analysis* – this suggests an analysis with a high correlation to prior knowledge within the field. This analysis method is related to the systematic literature review and is not how the results have been analyzed during this project [18].
2. *Empirical analysis* – this suggests an objective analysis in regards to the resulting data. For a purely objective analysis to be viable, objective data has to exist. This can, for example, include the time difference for execution when it comes to the results of a redeveloped algorithm. The results of this project were more subjective in nature, and empirical analysis of the results was therefore not suitable [19].

2.8 - The project's Scope and Limitation

The scope for the project was to produce a tool so that:

- It employed the researched 2fa and honeytoken techniques so that the tool was functional and could run without user interaction.
- It was only a prototype for exploring the proposed technique's feasibility. However, discussions on how the technique could be improved in order to make it suitable for the real world can be found in *Chapter 5.3* and *Chapter 8*.
- It integrated 2fa, honeytokens, and misinformation defense in a deceptive manner.

Limitations of the project were that:

- It would be limited to the focus of masqueraders. Other kinds of unauthorized entities would not be a focus.
- Testers would all be aware of the basic premises of the tool, and the testing environment would be isolated. It would not be tested “in the wild.”
- It would only be developed for and tested in Linux.
- It would only be developed for SSH shell logins(non-GUI-compatible).

3 – Theoretical Background

In the section's *background (1.1)*, *Related Work(1.2)*, and *Problem Formulation(1.3)* of *chapter 1*, a general overview of the project's background was established. During this chapter, concepts only mentioned briefly in these previous sections will be elaborated upon in detail. In particular, the chapter will cover details regarding honeytokens, 2fa solutions, and misinformation applications. In addition to this, the value of integrating different techniques and the concept of cyber-security deception will be explored. A general understanding regarding intrusion detection systems is expected on the reader's part and will therefore not be provided. The theoretical background is a result of the limited literature review (see *Chapter 2.4.4)*.

3.1 – Honeytokens

The honeytoken concept is an extension of the concept of honeypots. The honeypot concept derives from early 1990s reports regarding how deception can be employed to learn about an attacker's intentions [20]. They both seek to deceive intruders, but the difference between a honeytoken and a honeypot is that while a honeypot represents a complete computer system, a honeytoken is a more limited part of a computer system (for example, a file with misinformation) [21]. The first academic mention of honeytokens occurred in de Barro's report from 2003 [22]. The name "honeytoken" can sometimes prove a bit deceptive since a honeytoken can be more or less anything and fulfill a wide range of use-cases. Srinivasa et al. stated that a honeytoken could be designed in many different ways, and lists for example: [23]

- Canarytokens [24] – a honeytoken that serves the purpose of notifying admins that the system might have been compromised. It can be, for example, a link or file that is clicked.
- Honeybits [25] – these are very limited honeytokens, which together represent a deceptive trail to a destination (for example, a honeypot) that the defenders would like to lure the attacker towards. It can, for example, be fake log entries.
- HoneyLambdas [26] – this is a URL honeytoken that leads to a fake HTTP/S endpoint. It can, for example, be implemented in documents

or emails.

- Honeywords [27] – this is a honeytoken password. If someone logs in with this password, an alert is raised.
- Honeyfile [28] – this is the most generic variety of honeytoken: A fake file. It can be more or less any file.
- Honeyentries [17] – this is a fake entry into a database.
- Honeyaccount [17] – this is a fake user account that should not be used. If this account becomes active, then it indicates a compromised system.

Now that it has been established what a honeytoken can be, one can see that it is possible to use the concept in many ways. For the technique proposed in this project, 2fa honeytokens were employed.

3.2 – Misinformation

Misinformation is information that aims to "deceive people into forming an opinion, making a decision, or causing damage to an organization or group of people" [29]. The concept is generally nontechnical and has spread widely within public consciousness after social media overflowing with it during recent years [29]. If a source of information is incorrect or incorrectly reported, then the chances are high that it should be classified as misinformation. In this project's case, the focus is on defining cyber misinformation. As stated above, cyber misinformation is used to deceive other entities [9]. The scope for cyber misinformation can therefore include any technique which seeks to deceive its recipient. This includes, for example:

- Phishing emails
- Fake product information
- Fake emails between company employees
- Fake videos/audio (arguably a very severe form of misinformation)
- Misleading news
- Propaganda

Note that the above examples are the author's unreferenced examples.

One scenario in which misinformation can be used is when attempting to trick intruders into exposing their methods and intentions. Consider, for example, a file system that the intruder has gained access to. This file system is populated with data that seems real but is, in fact, misinformation. If the intruder cannot determine that the file system is a honeypot, then their actions within this system will reveal their capabilities, intentions, and so forth [9].

The field of misinformation is extensive, but for this project, the above information should be sufficient. To provide a technique that allows for the implementation of misinformation is an objective of this project. But, as stated in previous chapters, *how* to implement and use misinformation is not an objective. However, efforts to better understand modern forms of cyber misinformation are receiving grants from governments and governmental organizations, among them the E.U [29]. This funding is seemingly the result of the large-scale influence operations occurring on social media, and the aim is to understand how content deception functions and can be counteracted [29].

3.3 – 2FA

The concept of 2fa spans a wide array of available techniques. Any kind of additional method for verifying access can be added to the list of 2fa techniques. If two individual methods ("keys") for authorization are used together as a complete technique, then this is a 2fa solution [3]. In this section, different use-cases for 2fa will be detailed, together with some of their strengths and weaknesses. The 2fa concept is, however, not a core aspect of the project. The use of honeytokens for fulfilling 2fa functionality is the core aspect.

When outlining different factors that can be employed in a 2fa solution, Papaspirou et al. mentioned the following: [3]

- Knowledge Factor – this is something that the user has in their own mind, for example, a password. The factor is knowledge-based. The knowledge factorial can be an OTP (one-time-password) sourced from, for example, a 2fa application.
- Possession Factor – this is something that the user has in their possession. The factor is object-based (for example, an ssh-key or a Ubikey).
- Inherence Factor – this includes something that the user is, such as fingerprints or biometric data. This factor can also be voice or keystroke speed-related.
- Location Factor – user can, for example, only log in from within the external network or from a certain computer.
- Time Factor – user can only be logged in during a specified period of

date/time.

Commonly, 2fa solutions make use of one of the three first factors. However, an evolved method of 2FA is called MFA (Multi-Factor-Authentication) and more often uses the location and time factors.

In summary, a 2fa solution combines two of the factors mentioned above to authenticate a user's attempt to authorize themself. A caveat to this is that two factors from the same category can not constitute a 2fa solution. Factors must be chosen from different types [3].

The 2fa technique is therefore related to access-control. The access-control concept concerns restricting data to only those who are authorized to access it [30].

3.4 – Deception

The core of cyber-security deception is to increase a system's entropy by "simulation (showing the false) and dissimulation (hiding the real) techniques." [9]. The focus for this project when it came to deception were two aspects:

- The 2fa honeytokens
- The misinformation defense

When it comes to the use-case for cyber-security deception, the concept is mainly used to trick operators into believing that they have gained access to a real system and subsequently monitor them and gather knowledge regarding their intentions, tactics, and general objectives. Ahmad et al. states that if deception is successful, an intruder will be fooled into believing that their operational security is intact, when in fact, it has been compromised [9]. The points in the bullet list below, are benefits of deception: [20]

- Adversaries waste time and resources in a fake system. They learn nothing of value and instead might give away their information. It is also possible to feed the harmful intruder misinformation.
- It makes the potential intruder more risk-averse if they know about the deceptive defenses. A deceptive system may therefore deter intruders from even initiating an attack.
- Offensive attacks are slowed down or completely stopped. Since more systems are active, more data must be processed to understand the

complete network/system.

- Potential to analyze an intruder's tactics in real-time.
- It represents an additional technique for detecting intruders and can be employed with other techniques to provide a complete defense [31].

When considering these benefits, it should be clear that deception provides the defenders of a system with more or less limitless options regarding how a system should be protected. While individual techniques themselves provide the framework, deception is the strategy for how they function. Consider, for example, the game of Chess. The various pieces incorporate different functionality, but the player organizes them in different formations and tries to outwit the opponent. So it is with the concept of deception also in cyber-security. The various actors try to fool each other into making mistakes that expose the opponent's intentions and capabilities, with the end goal of check-mating the king (sysadmin) and taking over the board (system).

In general, the measures available for protecting a computer system does not individually provide a strong enough defense. Instead, they need to be used together to provide a complete defense [32]. Faveri et al., for example, stated that "traditional measures, although essential in any modern security arsenal, cannot provide a comprehensive solution against Internet threats." [20]. Deception is one way to move beyond these traditional measures. A defense with deceptive capabilities is most effective when individual techniques are used in combination [33].

The following examples portray how honeytokens and misinformation can employ deception:

- A honeytoken with a backdoor installed is perceived by an intruder to be real. The intruder downloads it and opens up their own system to analysis. Since the deceptive qualities of the honeytoken were used to fool the intruder, the honeytoken has fulfilled its use case. Honeytokens in themselves are deceptive mechanisms, so they are a natural addition to a deceptive system [9].
- A honeytoken is hidden away in a secure location. This honey token is filled with valuable information, with the caveat that one of the algorithms is defective. An intruder that compromises the system deep enough to access the honeytoken might find it interesting and

bring it back to those responsible for the operation. Those responsible hand the honeytoken misinformation over to an expert. The expert then determines the validity of the information. The expert is not able to notice the defective algorithm, and the honeytoken misinformation is therefore used. A pipeline is built and explodes.*

Finally, Han et al., for example, discussed using deception to monitor a general threat landscape or gather information about intruders. Concerning this, they also stated the definition for *Intrusion Deception* [34]. The intrusion deception concept only includes deceptive methods involved in detecting intruders. Due to the technique proposed in this paper relying on deception for detecting an intruder, the intrusion deception concept is of relevance to this proposed technique.

*"During the cold war, the CIA allowed a KGB team to steal manipulated data. This eventually lead to the trans-Siberian gas pipeline exploding." [9]

3.5 – Countermeasures

When a potential threat has been identified, the defensive systems must respond to this by employing a countermeasure. When it comes to countermeasures, many choices exist, but for example, Fraunholz et al. made use of the countermeasures presented in *Table 3.1* [17]. These responses were active since the program had to do something when suspicious behavior was detected. Also, the system isolation was an active response since it was not active by default. In addition to countermeasures that are activated, there also exist passive responses that are active by default.

The technique proposed in this paper employed two countermeasures. The main one was to limit access to the folder with valuable data. The second was to log data about the potential intruder, and notify the admin.

Name	Priority	Implementation
User alert	1	*wall [message]*
Admin alert	2	*UDP* alert to admin *PC*
System isolation	3	*iptables* reject all incoming
Mitigate exfiltration	3	*iptables* reject all outgoing
Evacuate sensitive data	4	Linux *shred [sensitive files]*
Self-destruction	4	*shred [filesystem]*

Table 3.1 - Countermeasures implemented in their deception framework - Fraunholz et al. [17]

3.6 – Inotify Hooks

Inotify – the technique used for surveillance of the 2fa honeytoken – is a file change notification functionality within the Linux Kernel [35]. Inotify sets up a kernel hook towards the specified file/files/part of the filesystem. Various flags can be specified, indicating when the hook should trigger a response – for example, when a file is changed or simply opened. When the hook is triggered, the program that created the Inotify hook responds in a specified way – for example, shutting down the system or simply logging user info. Since Inotify is hooked into the Kernel, it does not need to constantly scan the protected file but is informed when access occurs. This limits the resource usage of Inotify to a bare minimum and stands in stark contrast to using, for example, a lsof scan for monitoring. However, Inotify is not suitable for filesystem protection only due to the low resource usage. For example, Srinivasa et al. stated that for an intruder to find out that Inotify is running, they need to access the system's background processes. If such a process can be found, the intruder can assume an "alerting mechanism" is active [23]. When Inotify is run from root, an intruder has to access root processes to identify the potential alerting mechanism. This "invisibility" enables a high degree of deception (the system has to be compromised to the root before information regarding Inotify alerts can be detected).

3.7 – Research Gap

The scientific gap explored during this project was whether the proposed technique should be considered a relevant defensive technique. The

misinformation defense was not part of the exploration gap but simply an aspect of the complete technique. The proof presented that this research gap exists is that this technique was not mentioned in the surveyed literature. By exploring the various reports, while also searching on keywords such as “2fa defense”/“2fa misinformation”/“2fa intrusion detection,” it was not possible to find any articles with proposals resembling the challenge of this degree project. It seems that 2fa functionality generally is used as a technique in itself rather than as an integrated aspect of a complete technique.

3.8 – Theoretical Background Conclusion

By analyzing the previous sections of this chapter, one might deduce that the techniques are related to one another: they all seek to protect valuable information. For example, a honeytoken can be filled with misinformation and placed in a deceptive system area. An intruder that finds the misinformation honeytoken is hopefully satisfied with extracting this honeytoken instead of exploring the system further and finding a file with real information. A honeytoken can also be more or less any artifact that seeks to expose potential intruders in one way or another. Because honeytokens are such a highly moldable concept, it also seemed reasonable to explore if they could be used as 2fa tokens.

Finally, one reason this project’s research may be considered motivated in regards to the theories presented in this chapter is that Han et al. state that “traditional measures, although essential in any modern security arsenal, cannot provide a comprehensive solution against Internet threats.” [24]. In addition to this, Zohar et al. concluded that honeytokens are useful additions to a defensive system but that frameworks which “combine and automate measures are scarce” [32]. In general, the opinion within the cybersecurity community seems to be that various techniques need to be integrated for them to be truly effective. For example, Han et al. state that traditional measures can not provide a comprehensive defense by themselves but need to be integrated with deceptive measures [34]. Concerning this, Fraunholtz et al. argued that a deception-based defense should be made up of various integrated techniques [33]. Similar opinions regarding the integration of different techniques have been stated for a long time. For example, Stolfo et al. proposed that individual techniques should be integrated, resulting in a more complete defense [8]. Ahmad et al. stated that a key component of a

misinformation defense being successful is that the misinformation is presented so that the intruder believes it to be real. To achieve this, they argued that different system elements must be integrated [9]. As can be seen, many researchers stress the importance of integrating techniques to achieve a more complete protection. Integrating 2fa, honeytokens, and misinformation in a deceptive fashion was a goal of the project.

4 – Research Project Implementation

This chapter will present a detailed overview of the implementation created during the DS process described in *Chapter 2.3*. This includes a diagram of the tool's execution algorithm and a bullet list describing this algorithm's various stages. This chapter intends to provide the reader with an understanding of the implementation itself and what functionality was included to fulfill the feasibility requirements checklist of *Chapter 2.4.2*. Foremost, the implementation was used in order to answer the checklist requirements of RQ 1.2 (Could technique fulfill a large number of feasibility-related aspects). However, the implementation was also used to enable the user-testing of RQ 1.1 and to answer the complete RQ 2 (regarding how the technique compared to prior techniques). Finally, the implemented technique was showcased to the cyber-security expert(RQ 1.3). The implemented technique was the DS artifact used to prove the feasibility of the proposed technique. It was the result of the iterative development process (see *Chapter 2.3)*.

The basic concept of the technique was adding a 2fa honeytoken file to a user's filesystem. This file was only intended for access by the user who owned it. When the user had logged in, they only had access to a fake folder with misinformation. To be granted access to the correct folder with the real information, the user had to access the 2fa honeytoken within 30 seconds. If the user did not do so, an alert was triggered. The alert sent a warning to an admin. If the user accessed the folder within the time limit, they were granted full access rights to the correct folder and could begin working. The program repeatedly – every 5 seconds – ran a scan to see if the user was still logged in. If the user was no longer logged in, the user's access to the correct folder was once again removed, and again only the fake folder could be accessed. In order to regain access to the correct folder, the user, therefore, had to access the 2fa honeytoken once more. For a succinct overview of the functionality described above, see the flowchart presented in *Diagram 4.1* and the algorithm description below the diagram.

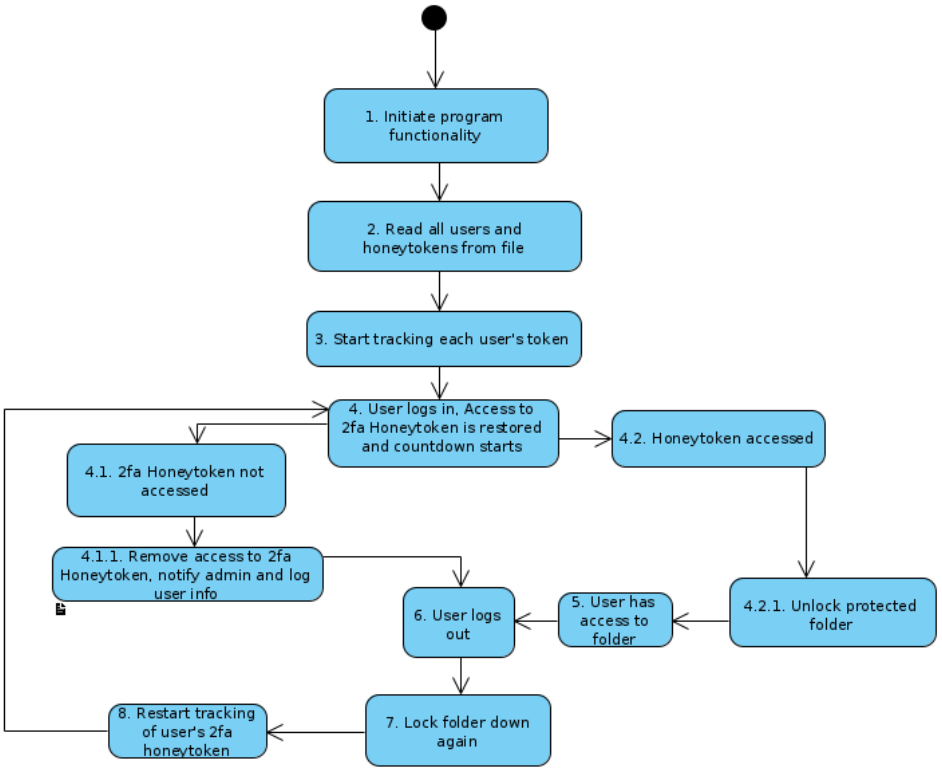


Diagram 4.1 – presents an overview of the implemented technique's various stages.

The below pseudo algorithm provides an overview of the functionality presented in *Diagram 4.1*.

1. *Program's 2fa honeytoken tracking functionality is initiated.*
2. *Reads file with all users and 2fa tokens.*
3. *Monitoring of these files and users is initiated.*
4. *User logs in → a countdown of 30 seconds begins (the folder is hidden).*
   4.1. User does not access 2fa token → Admin alerted, and only fake folder remains available. User's info logged (IP, Time, Etc.) → 4.1.1 to activate the protection mechanism → go directly to step 6.
   4.2. User accesses 2fa token within time-limit → 4.2.1 to unlock the protected folder so that user can gain access to it.
5. *The user is provided with access to the folder.*
6. *Program monitoring picks up that the user is no longer logged in →*

returns to step 3 to monitor the user for login again.

7. *Locks user's folder down after logout -> Program still runs the monitoring of 2fa tokens and users until it is shut down. Users can log in and out without complications.*
8. *The program starts tracking the logged out user's 2fa honeytoken for activity again. When the user logs in again, the countdown is initiated again.*

The intention behind implementing the technique described by this pseudo-algorithm was to create a technique that provided a passive means of defense. Consider the overview of countermeasures that Fraunholz et al. made use of during their project (refer back to *Chapter 3.5* if necessary). Their countermeasures were active in nature, but the countermeasure proposed in this project was mainly passive. The protected data was protected until – at least this was the intention – suspicious behavior was ruled out. The tool did not wait for the masquerader to expose itself after login and then remove access (an active response). It never provided access in the first place (passive protection). In doing so, the masquerader (1)could not find suspicious activity due to the system actively responding to suspicious behavior, and (2)could not access the valuable information. One might argue that a passive mechanism is a relevant addition to the more active mechanisms presented by Fraunholz et al in *Chapter 3.5*. Consider, for example, a firewall for a more secure network. Instead of allowing packets into the network until they are detected to represent something unwanted, they are never let in at all but rather silently dropped. Only packets explicitly approved are allowed to pass. The core of this project's proposed technique is to – in a similar fashion – silently remove the threat that unwanted masqueraders present. To see how well the implemented technique fulfilled the requirements checklist, see *Chapter 5.2*.

5 – Results and Analysis

In this chapter an overview of the results gathered will be presented and analyzed. This chapter aims to present the reader with detailed results and analysis of the usertesting and the checklist. For the results of the limited literature review, see *Chapter 3 & 6.3*.

5.1 – RQ 1.1, Usertesting and Interviews

In the section below, the testers' responses to the interview questions are presented, and their answers analyzed concerning the potential feasibility of the technique. In addition to this, the testers' general opinions on the technique, and their suggestions for improvements, will be presented.

5.1.1 – Interview Results and Analysis of Testers' Answers

This section will analyze the testers' answers to the interview questions according to the methodology presented in *Chapter 2.4.1*.

1. *Do you think this might be a relevant technique to develop further?*
   Results: Only one of the testers (tester L) had initial reservations regarding this. The reason was that the 2fa token represented a potential liability. However, when tester L better understood how the token functions, he considered the technique worthy of further development. The initial response of the other three testers was a clear "yes."
   Analysis: Since all of the testers considered the proposed technique relevant and worthy of further development, this was considered as strengthening the case for its feasibility.
2. *Was the 2fa honeytoken irritating to use?*
   Results: None of the testers had any problems with using the 2fa honeytoken and protected folder. The information they were provided with (found in *B. Appendix – Information for Testers)* generally had provided them with a good understanding of the basic premises of the technique. During live testing, they seemed to pick up on how the technique worked in reality easily.
   Analysis: Since none of the testers had any problems understanding how the token functioned or how to use it, their answers to this question were considered to strengthen the general feasibility of the

technique. Due to the proposed technique's heavy reliance on the 2fa token, the testers' answers to this question were considered especially relevant regarding the general feasibility.

3. *Potential weaknesses of the technique?*
   Results: The testers' main reservations were in direct relation to the 2fa honeytoken. For example, if it would not be hidden well enough, or if the time limit would create problems. For example, tester D stated that: "The time limit is good, but if the user simply forgets, then it becomes a waste of time for the admin.". From the testers' point of view, the tool's main potential weakness was related to different aspects of the 2fa honeytoken.
   Analysis: The testers' answers to this question were closely related to the topic of question 2 since all of them also mentioned the 2fa token to be a potential weakness. Since the 2fa token is such a vital part of the technique, their opinion of it being the most obvious potential weakness makes sense. If the token would become compromised by a masquerader, the value of the proposed technique would suddenly become null. A detail regarding this is that their answers to this question could be considered to strengthen the assumption that the testers truly understood how the technique functioned. They could discern the core aspect, which, if compromised by an intruder, would remove the protection completely.
4. *Where would you hide the 2fa honeytoken?*
   Results: Answers to this question varied greatly, with the common denominator that the token should be hidden somewhere it would not be expected. Tester D provided an interesting point of view, stating that the tool should be hidden in different places depending on the scenario. For example, an individual should probably hide the token in a different location than a company should. The general opinion of the testers – was that the honeytoken could be hidden more or less anywhere in the system, as long as this location is one where an intruder would not look for it, or by accident, come across it.
   Analysis: The testers generally agreed that the token should be placed somewhere that it would be unlikely for it to be found by random by someone unaware of its functionality or by intent by someone aware of the technique. What was surprising was that the tester also came up with relatively detailed ideas on how this should occur. The fact that they were able to do so is argued to strengthen the proposed

technique's feasibility. If users can come up with locations to hide it, while at the same time these locations vary from user to user, this could imply that the 2fa token can be well hidden in unique locations by the users themselves. This reasonably improves both the protection of the 2fa token and the user's potential to remember where it is located (since they have themselves decided the location). At the same time, the workload of the admin is lowered (since they do not have to spend time thinking of new places to hide the tokens). If the admin would have to decide the location of all 2fa tokens, this could lead to a less diverse number of locations where they would be hidden.

5. *Potential strengths of the 2fa honeytoken?*
   Results: The main point was that three of the users considered accessing the 2fa token after login a strength.
   Analysis: Since testers mentioned the fact that the 2fa token is accessed after login, and this aspect is part of the proposed technique's core functionality, this was seemingly an important opinion to take into consideration. It further strengthened the argument that the testers truly understood the technique itself. They were able first to discern the potential main weakness of the tool (in response to question 3) and then, in response to this question, the main strength of the tool. The fact that the testers seemingly understood the technique should reasonably lend greater relevance to their opinion that the technique is relevant (as they stated in response to question 1). If the testers could not prove that they understood the functionality of the technique, their opinion on its feasibility would be of limited relevance.
6. *How would you design the honeytoken?*
   Results: This question did not lead to relevant answers. Two of the testers had nothing to mention, and the two other testers suggested implementing a password.
   Analysis: A more in-depth discussion than testers could provide would have been interesting, but it was not possible to have one. The reason testers were unable to discuss this question in a relevant way can perhaps be found in how this question was more technical than the other ones. Yes, testers understood how the tool functioned and could point out various relevant aspects of it. But this question was more concerned with developing the technique further, and for that

kind of question to be discussed, testers would probably need time to think about the tool for some time after testing it. The results from this question are therefore not relevant concerning the feasibility of the technique.

5.1.2 - Testers' General Thoughts and Suggestions for Improvement

One interesting aspect of the testing was that the testers readily suggested potential improvements or alternative ways of using the 2fa token. For example, tester R stated that" I think that if you could make it so that the place where people login is a jump server, and then when the 2fa is accessed, the user can access the real ssh key." Tester R came up with a new use case entirely on his own, which is relevant concerning the technique's feasibility. It suggests that using the 2fa tokens in many different settings should be possible. This is especially important for the aspect mentioned in *Chapter 3.8* concerning the integration of different techniques. If a user easily can develop new use-cases for the technique, then it would be possible to integrate it into many different environments. Tester D built further on this idea, stating that if implemented as a jump server protection, a fake .ssh folder would need to exist for the tool's core functionality to remain intact. Tester D also suggested that the ssh keys in the unprotected folder would lead to a honeypot. Tester D also suggested that linking the 2fa token to specific hardware would further strengthen the protection without adding, for example, a password to the 2fa honey token. The main takeaway for this topic is that the testers found the technique interesting and relevant. Since they could discuss relevant improvements on their own accord, they probably did not only say it was relevant since they wanted to be kind

5.2 – RQ 1.2, The Tool's Checklist Compliance

In this section, the results and analysis of the final iteration of the checklist will be presented. See *Chapter 2.4.2* for details on the methodology used for gathering the tool's checklist results and *Chapter 4* for the algorithm describing the tool's execution process.

5.2.1 – Analysis of the Checklist

In this section, the results of the checklist are analyzed. Whether or not the individual results should be considered as strengthening the technique's feasibility will also be stated.

1. *Always responds to 2fa token activity?*
   Result: The tool always responds to 2fa token activity.
   Analysis: since the Inotify ran as a Kernel hook, the program always responded to the 2fa activity. Inotify is a core functionality of Linux. The functionality Inotify provided the tool with was reliable and used resources sparingly due to hooking to the kernel and waiting for access to occur, rather than constantly scanning for such access. No identified weaknesses. The tool used the Inotify kernel system. Since the tool was run from root, these processes were isolated.
   Strengthens case for feasibility: Yes. Reliable.
2. *Always locks/opens folder?*
   Result: Always opened folder when token accessed, always locked when the user had logged out.
   Analysis: Yes. From the first iteration until the end of the final iteration, the program always handled this functionality as intended. However, instead of a protected folder, the users should log in to a virtual machine. When they access the 2fa token, they would be let out of the VM and into the real system. One more specific weakness is that during the 5-second scan to see if the user is still logged in, the masquerader could log in quickly after the authorized user logs out, and then the access to the real folder would remain.
   Strengthens case for feasibility: Yes. Reliable.
3. *Time limitation is functional?*
   Result: Yes, the time limit generally worked as was intended.
   Analysis: The time limitation, once implemented, undermined the program functionality when users with similar names, such as “test” and “est” were active on the system at the same time. This resulted from one of the Bash commands run by the Python program, leading to a scan for “est” picking up “test” if he was logged in. If users did not share names in such a way, no problems occurred.
   Strengthens case for feasibility: Yes. The tool’s main reason for development was to prove the feasibility of this kind of 2fa honeytoken, not how to scan correctly after users.
4. *Testers can use the program easily?*
   Result: Yes, no one had any problems understanding how to use the 2fa honeytoken.
   Analysis: Testers easily understood the program. By reading the document (located in *C. Appendix – Information for Testers*), they

gained a good basic understanding of the tool. When testing occurred, they had no problems using the functionality correctly.
Strengthens case for feasibility: Yes, but extra iterations of user-testing should have been incorporated in order to further explore this aspect. The environment in which the testing took place would also have benefited from being more realistic.

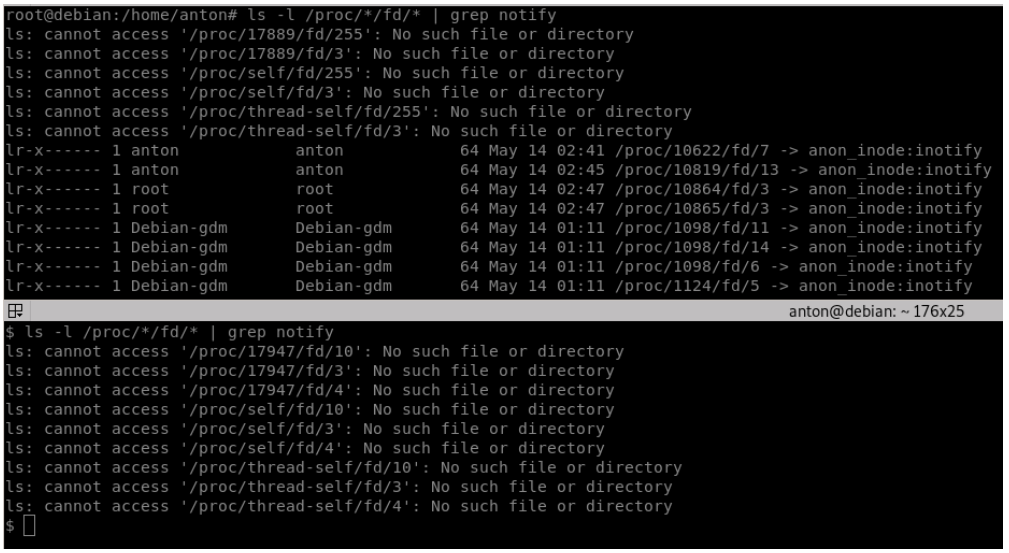
5. *Tool is not resource intensive?*
   Result: In its final state, the tool was not resource-intensive.
   Analysis: Generally, the tool used close to no resources. It simply set up an Inotify process for each honeytoken and then waits. The scan runs every 5 seconds, is a direct bash command and not a general file system scan. During the whole process of developing the technique, resource usage was never a problem - neither on the server with only 1 CPU core or on the laptop stated as the context from which to determine resource intensity.
   *See picture 5.3 at the end of this section.
   Strengthens case for feasibility: Yes. This technique can be implemented on very basic systems without straining resources.
6. *Cyber-security expert says the idea is feasible?*
   Result: Yes.
   Analysis: The cyber-security expert stated that the "technique in itself is relevant." This is important since a cyber-security expert has a good understanding of the overall research field.
   Strengthens case for feasibility: Yes. But the cyber-security expert mentioned that further research needs to be done to determine the extent to which the technique is relevant.
7. *Logs user info when 2fa is not accessed within timelimit?*
   Result: Yes, ran a "who -i | grep username", logging IP, date, etc. Also logged the user's active processes.
   Analysis: Since the tool is a monitoring tool, it needs to log suspicious behavior. It, however, only does so one time. It should track the activity of suspicious users. Only logged relatively limited activity info (open processes).
   Strengthens case for feasibility: Yes.
8. *2fa honeytokens and users can be added automatically?*
   Result: Yes and no. Honeytokens could be created from within the program when a new user was added. But it was not possible to let the tool read from a list and create all of the folders, honeytokens, and

users.
Analysis: Since the automation of the tool only concerned the 2fa honeytoken, while it also had to be automatically created when running the tool, this is a limitation.
Strengthens case for feasibility: Currently no. This aspect needs to be developed further if it should be argued as strengthening the case for feasibility.

9. *Warning is sent to admin if 2fa honeytoken is not accessed within the time limit?*
Result: Yes, a ”send-notify critical Intrusion-warning” was sent to the GUI of the user running the program.
Analysis: Since this only occurs if the user running the tool has a GUI, it is a severe limitation.
Strengthens case for feasibility: No. Should be able to send a notification to the terminal/email/phone etc.
10. *Tool can run with several users active at the same time?*
Result: 50/50. The time limit functionality led to a 30second while-loop, during which other users could log in without being noticed.
Analysis: The tool’s multi-user functionality strongly adds to the feasibility of the technique, since a technique that can only protect a single user would serve a very limited purpose.
Strengthens case for feasibility: Yes.
11. *Does not need to be restarted?*
Result: It did not need to be restarted. Unless new users were to be monitored, in which case the tool had to restart so that the user registry file was reread.
Analysis: Could run without restarts no matter how many times users logged in or out. It would be unreliable and become more of a liability than protection if it could not do so.
Strengthens case for feasibility: Yes, but it needs to be run in real-world conditions to explore this further.
12. *Researched how Inotify can be circumvented, and implemented potentially necessary protection against this?*
Result: Regular users could not access the /proc folder. Therefore the Inotify processes were hidden from them.
Analysis: Since the Inotify toolkit is well known and has very specific functionality, a masquerader who could circumvent it would

gain great power over the protective systems. But since the tool is run from the root user, and regular users do not have access to the /proc folder that handles the Inotify processes, a masquerader can't access Inotify processes without first gaining root access.
Strengthens case for feasibility: Yes. This was a very important aspect of determining feasibility. If users could easily access the Inotify processes, then they could perhaps also manipulate them somehow. Since they cannot access them in any way, not even being able to see if they exist, this makes the protection invisible to an intruder.
*See pictures 5.1 and 5.2 below.

13. *Remove command line history, so that a masquerader can't find the 2fa honeytoken in the .bash_history?*
Result: The .bash_history file in the user's /home/username was removed after logging out. The program only tried to delete this file if it existed.
Analysis: This was important for the feasibility case since if the history was saved in between sessions and the masquerader accessed it and saw the 2fa location, it would not matter where the 2fa honeytoken was hidden.
Strengthens case for feasibility: Not really. This was such a simple thing to implement. At the same time, if logs holding the location of the 2fa honeytoken do not exist, this makes it hard for a masquerader to find the token within the 30-second time limit.



Picture 5.1 – Inotify Process Access Example #1 – Only the user root can access the Inotify process information

Picture 5.2 – Inotify Process Access Example #2 – Only the user root can access the Inotify process information

As was stated in the results and analysis above: Johan (being one of the tester accounts) can not access information about the Inotify processes. Root, however, having access to the complete system, can.

Picture 5.3 – the technique's resource usage was limited

When the tool ran, tracking five users for login/logout, it used 0.0-2.5% of the laptop's CPU - generally resting on 0.7%. The static memory usage was 0.8%. These numbers were not considered to be resource-intensive (see *Page 16* for how "resource-intensive" was defined*)*.

5.3 – RQ 1.3, Cyber-security Expert's Feedback regarding Technique's Feasibility

For details on how the final result was showcased to the cyber-security expert, see *Chapter* 2.4.3. The cyber-security expert's feedback was that "the technique in itself is relevant," but various aspects needed clarification. These aspects were:

1. *Protection of the Folder* – What happens if an intruder logs in when the authorized user has opened the filesystem? If this would occur, then the intruder would have complete access. Due to this, it was a severe weakness of the current implementation. The tool was also designed not to activate the countdown again if the user was already logged in (thus removing the admin notification protection). This risk could perhaps be mitigated by sending a notification that a user has logged in two times and removing access to the protected folder and 2fa honeytoken. An additional point raised in regards to this can be found in *Chapter 5.2.1*. See point 2 of the checklist.
2. *Scanning for the file* – How difficult would it be for an insider who is aware of the functionality to find the file? This depends, but The file was left without a file-ending to make it more generic. However, this could also be a weakness since scanning for files without file-endings would be possible. But an insider could perhaps wait for the countdown to run out, then scan for all files with completely locked permissions, and then relog and access the now unlocked 2fa honeytoken. The insider would thus gain access to the locked folder. Locking down the 2fa token completely after the time limit has run out might therefore be a weakness.
3. *The Misinformation Protection – How effective is it? How complex does it have to be? How would it affect the general functionality, such as resource usage?* These aspects are all hard to evaluate from the point of view of this project. However, as was stated in C*hapter 3.7*: "The misinformation defense was not part of the gap to be explored, but simply an additional aspect of the complete technique." Therefore, the focus has only been on how misinformation could be presented and how to integrate it with the 2fa honeytoken. During this project, the misinformation was placed in a fake folder with important files. It would not be possible to answer these questions adequately without further research. For more on this subject, see the final paragraph of

Chapter 8.

4. *Attacking the Tool* – What would be necessary for a successful attack of the tool? Could, for example, a regular user see which 2fa honeytokens are monitored? No, a regular user could not access information about the inotify processes. For a more detailed answer to this, the reader is referred to the section *Analysis of the Checklist,* found previously in this chapter. See point 12. Concerning this, see *pictures 5.1 & 5.2.*
5. *What would happen if the tool was stopped?* – If the tool was stopped, the system's current state remained until the tool was started again. If the tool was stopped before the 2fa honeytoken being accessed, the user would have to use sudo functionality to unlock the protected folder. If the tool were stopped when the folder had been unlocked, then the folder would remain unlocked, providing free access to a masquerader. If an attacker can stop the tool when the user has unlocked the folder, the protected files become vulnerable.

6 - Discussion

In this chapter, the findings of the project will be discussed. This mainly concerns a discussion on the validity and reliability of the various results. The results will be tied to their respective research questions and then discussed. In addition to this, the results of the project will be set in relation to prior research. This chapter only focuses on the reliability and validity of the research question results. For the actual answers to the research questions, see *Chapter 7*. For the complete method overview, see *Chapter 2.4*, and for the plan to improve the reliability and validity of the results, see *Chapter 2.5*.

6.1 – RQ 1, reliability and validity

A discussion to answer Research Question 1, which concerned the general feasibility of the proposed technique, included the user-testing, the requirements checklist, and the cyber-security expert's feedback. For details on the individual methods, see *Chapter 2.4.1 & 2.4.2 & 2.4.3*. For the results they led to, refer back to *Chapter 5*.

6.1.1 – RQ 1.1, Could users interact with the technique successfully?

During this section, the reliability and validity of the user-testing will be discussed. This was the method for gathering results to answer RQ 1.1. See *Chapter 5.1* for the results.

Were the interview questions of good quality? Testers were generally able to understand the questions and provide relevant answers. Since the questions did not confuse testers and resulted in clear answers, they were considered reliable.

What knowledge of cyber-security did the testers have? When determining the validity of the user-testing interview results, it is worth considering that the testers were students and had limited knowledge of the field of cyber-security. The results of the user-testing phase were therefore only validated within the context of computer-science students.

Did the testers exhibit a good understanding of the technique? During testing, testers were able to use the technique and discuss potential strengths and weaknesses readily. Since testers exhibited a good understanding of the technique, it is argued to strengthen the reliability of their answers.

How extensive engagement did the testers exhibit? When it came to the testers' engagement with the technique, they seemed to be genuinely

interested in it and were willing to spend time discussing and exploring the technique. They often gave extensive answers to questions and seemed interested in discussing potential improvements (the exception being tester J). Due to the high engagement of the testers, this was considered to strengthen the reliability of their answers.

Aspects that may have undermined the reliability of the testers' answers are that testers were friends and classmates, which might have biased them. A crucial aspect of the user-testing validity was that user-testing only occurred during the second development iteration. Additional testing iterations conducted over more extended periods would have added relevance to the validity and reliability of the user-testing results.

Overall, the tester's answers were considered to be reliable. The answer to RQ 1.1 itself was therefore considered to be reliable.

6.1.2 – RQ 1.2, Could the technique fulfill a large number of feasibility-related aspects?

During this section, the reliability and validity of the requirements checklist itself will be discussed since this was the method for gathering results to answer RQ 1.2. See *Chapter 5.2* for the results.

Were enough aspects related to feasibility included? Since the cyber-security expert stated that the "technique itself is relevant," it is argued that enough aspects concerning feasibility were included in the checklist. Something that would have improved the checklist's validity would have been if it was sent to the cyber-security expert early on in the project - and feedback on how to improve it was received.

Did the result of each aspect strengthen feasibility? Points 8 and 9 had in common that they would need to be developed further to strengthen feasibility reliably. They were not complex enough. Point 13 was also implemented in too simple a fashion. Instead of just removing .bash_history, more logs should be deleted. Doing so would more successfully validate that the 2fa honeytoken location remained hidden from intruders.

Were the aspects, in general, reasonably complex? Implementing the functionality necessary for validating the technique was not too complicated for the author. A complete tool could therefore be implemented and validated. The author understood how each aspect of the tool functioned and could therefore validate the results.

It should be possible to reliably recreate the checklist results, primarily since they do not depend on external circumstances, like being run

on a specific company's infrastructure. Overall, the checklist results were considered reliable enough for validating the initial feasibility of the proposed solution.

6.1.3 – RQ 1.3, Would a cyber-security expert consider the technique feasible?

The reliability of the results used for answering this RQ was considered high, since the results were derived from the opinion of a cyber-security expert. The cyber-security expert thought the proposed solution to be valid as an initial testing prototype but not for real-world use. Therefore, his opinion on the context validity of the project echoes the discussions of *Chapters 6.1.1 & 6.1.2*.

6.1.4 - RQ 1, Conclusion

The answers to research question one concerned isolated testing environments where the implementation was developed and computer science students interacted with the technique. However, as stated in *Chapter 2.5.2,* the project's results regarding feasibility would only be considered reliable if the cyber-security expert thought them to be so. Since the cyber-security expert did consider the proposal feasible, this is an essential point for arguing the reliability of the project. The cyber-security expert's opinion led to RQ 1.3 being answered and strengthened the reliability of the answers to RQ 1.1 & 1.2. Since the cyber-security expert also considered the technique feasible ("relevant in itself"), this lends credibility to the answers to research questions 1.1 & 1.2.

6.2 – RQ 2, reliability and validity

The method used for answering Research Question 1, which concerned how the proposed technique compared to already available techniques, was the limited literature review. For details on this method, see *Chapter 2.4.4*. For the main results derived from research question 2, see the comparison with prior research in *Chapter 6.3.*

6.2.1 – RQ 2.1, What were the advantages/disadvantages of this new technique compared to prior techniques?

The background research provided much information about the advantages and disadvantages of prior techniques, for example, how they functioned and

how to use them. Due to this, the research articles did provide valid information regarding the potential advantages/disadvantages of this project's proposed technique. For the answer to this RQ, see *Chapter 7*.

6.2.2 – RQ 2.2, Which use-cases might the technique be better/worse suited for?

When it came to potential use-cases for the technique, the background research provided examples of how and when to use different techniques. However, the examples were structured less efficiently than, for example, the advantages and disadvantages of prior techniques. Therefore, it was hard to find highly relevant articles discussing this subject, and it was necessary to make general observations from a wide range of articles and then draw independent conclusions. As referenced in *Chapter 2.5*, observations can be valuable preliminary results. Therefore, the answer to this question was considered relatively reliable — the validity of the answer was, however, limited to the author's research. For more extensive validity to be achieved, a paper that focused on the subject of cyber-security use-cases should have been included. By including such a paper, this would provide concrete examples organized systematically. For the answer to this RQ, see *Chapter 7*.

6.2.3 – RQ 2.3, Can the results of prior research be used to argue that this project's proposed technique is relevant?

Many highly relevant articles discussed aspects of the proposed technique and how to integrate individual techniques into more complete techniques. These articles provided relevant information for reliably arguing the relevance of the proposed technique. For the answer to this RQ, see *Chapter 7*.

6.2.4 – RQ 2, Conclusion

The results that led to this question's answer were considered valid within the context of an initial limited literature review (even if RQ 2.2 would benefit from further research). They were also considered reliable within such a context. The reason why they were considered to be reliable is that it was possible to find relevant peer-reviewed literature for answering the broken-down research questions. However, a systematic literature review could be the next step for a more extensive comparison to prior research.

6.3 – Comparison with Prior Research

This section will provide an overview of how this project relates to the main three articles of the background theory. The first focused on using honeytokens in conjunction with 2fa. The second focused on how a deceptive protection mechanism focusing on the misinformation distribution can be implemented. The third focused on how one can mitigate data theft insiders.

In their conclusion, Papaspirou et al. proposed a 2fa solution where honeywords (see *Chapter 3.1*) were used to expose potential masqueraders [3]. If a password flagged as a honeyword was used during their implementation, the compromise was immediately detected. The main overlap between the technique proposed during this paper and the technique proposed by Papaspirou et al. is that there in both techniques exists a honeytoken implementation that intends to expose an intruder without the intruder becoming aware of this. In the research made by Papaspirou et al, the intruder is provided with access when using the honeyword but is still detected. Similarly, the technique proposed in this paper also provides access to the system so that the intruder believes they have achieved real access. However, the technique proposed in this paper does not provide the intruder with access to the real system (The fact that no visible 2fa exists could, however, be suspicious and should probably be considered during future research.). The two techniques share common ideas and intentions. Concerning this, one might consider the statements quoted in *Chapter 3.8* regarding integrating different techniques with one another to enable a more comprehensive defense. One might argue that the technique proposed in this paper and the technique proposed in the paper of Papaspirou et al. are suitable for being integrated into a single, more complete technique. This could be done by adding honeyword protection to this paper's proposed technique.

The article composed by Ahmad et al. focused on implementing disinformation to protect valuable data from being stolen [9]. Their suggestions on implementing disinformation for this purpose were to do so in a deceptive fashion. As the reader perhaps recalls, the subject of deception was discussed during *Chapter 3.4* and is one of the core aspects of cyber-security protection. The relation between the paper of Ahmad et al. and this project's paper mainly concerns how to implement disinformation. However, the disinformation in itself – for example, the specific contents of the disinformation files - was not a focus of this project's implementation or

research. Concerning this, Ahmad et al. provided an overview of many ways in which disinformation could be implemented and used. Since they did not mention the kind of technique proposed in this paper, the addition of this technique to their paper may be of at least some value. During the conclusion, Ahmad et al., for example, states that their paper contributes to an overview of different ways in which an intruder may be fed disinformation in a deceptive manner. The core use-case of this project's proposed technique is to set up a protected folder on the filesystem (where the most valuable data is stored) while a "fake" folder with disinformation exists. The usage of 2fa honeytokens to implement the disinformation like that was not mentioned in the article of Ahmad et al. and may therefore be a relevant addition to it.

Stolfo et al. suggested monitoring user behavior to mitigate insider data theft. By doing so, it would be possible to detect a potential intrusion [8]. This included decoy documents being placed among real files. In relation to this user monitoring, Stolfo et al. explicitly mention masquerader detection as a potential pro of their technique. As mentioned, the technique proposed in this paper focuses exclusively on the context of masquerader detection, so the two papers share this aspect to at least a certain degree. However, the user behavior monitoring implemented by Stolfo et al. relied on the mentioned decoy files placed within the system. If one of these decoy files was accessed or used somehow, this indicated a potential intrusion. The honeytoken detection was, therefore, more active in nature. The intruder had to do something with the honeytoken for it to represent suspicious behavior. In comparison, the technique proposed in this paper is passive in nature. If the honeytoken is not accessed, then this represents suspicious user behavior. The two techniques are, in this sense, therefore both similar and dissimilar. Another point worth noting is that also Stolfo et al. raised the question of integrating techniques for them to become more successful in detecting intruders.

If the relation between the mentioned paper's and this project is still not clear, consider the following:

1. Papaspirou et al. used a 2fa solution together with honeytokens. This is also what this paper's proposed technique does.
2. Ahmad et al. presented an overview of deceptive techniques that can be employed to feed intruders misinformation. The technique proposed in this paper uses a fake folder with fake files, and through deception (2fa honeytoken accessed after login), seeks to fool a masquerader into believing they have access to ”real” data (thereby

feeding an intruder misinformation).

3. Stolfo et al. explicitly mentioned masquerader detection in their paper, and their implemented technique monitored user behavior to detect intruders. Both these aspects are implemented by this paper's proposed technique.

7 – Conclusion

This chapter will conclude the work as a whole, by presenting: the answers to the research questions; whether the technique was relevant (the research gap, see *Chapter 3.7)*; if the results might be relevant for other cyber-security areas than masquerader detection. Finally, a three-paragraph discussion concerning the project's potential contributions will be presented. First, the research questions will be answered. Refer back to *Chapters 1.3 & 2.1* for the complete details on the research questions.

- *RQ 1.1 Could users interact with the technique successfully?* – Users were able to interact with the technique successfully. They were able to understand how the tool worked and the problems that it intended to solve.
- *RQ 1.2 Could technique fulfill a large number of feasibility-related aspects?* – The tool was able to fulfill almost all of the feasibility-related aspects of the checklist. In the end, not all results were considered relevant for proving feasibility, but the great majority of points on the checklist were fulfilled.
- *RQ 1.3 Would a cyber-security expert consider the technique feasible?* The cyber-security expert clearly stated that the "technique in itself is relevant" and thought it worthy of further development, even if future developments would be necessary to make it feasible for real-world usage.
- *RQ 2.1 What were the advantages/disadvantages of this new technique in comparison to prior techniques?* – The main advantage of this technique over prior techniques is that it integrates 2fa and honeytoken functionality so that the user must log in to the system before accessing the 2fa token – and if this does not occur, the data is still protected. At the same time, the intruder believes it has real access. This tool aspect is seemingly new and relates strongly to the notion of deception discussed in *Chapter 3.4*. The main disadvantage is that the proposed technique seems to be more dependent on the user than is usual. Other techniques might only require the user to access a specific telephone number to proceed past the 2fa protection. This technique requires the user to remember a file hidden within the file system.
- *RQ 2.2 Which use-cases might the technique be better/worse suited*

for? – The technique is most relevant for a scenario where an organization would like an additional technique that integrates access-control with a countermeasure (see *Chapter 3.5)* while leaving the intruder unaware of this. Since the technique demands more from the individual users, it requires them to be motivated enough to interact with it correctly. The technique is unsuitable for a scenario where the users are unable or unwilling to interact with a relatively complex access-control tool. If the protected data is valuable, then it is a valid use case for the technique. If the data is not particularly valuable, it is probably not a good use case for the technique.

- *RQ 2.3 Can the results of prior research be used to argue that this project's proposed technique is relevant?* - A statement often repeated in the literature reviewed was that continuing to integrate different techniques is necessary to achieve a more reliable defense (preferably in a deceptive fashion – which this technique does). The proposed technique integrates various techniques in a seemingly new way and therefore incorporates this relevant aspect of cyber-security.

The answer to research question one (*is the technique feasible?)* is that the technique is feasible within an initial testing environment but needs to be developed further to become feasible for a real-world context. The answer to research question two (*how does the proposed technique compare to prior 2fa and honeytoken implementations?*) is that the technique takes prior 2fa and honeytoken implementations and seemingly adds a new twist to access-control when integrating them. The individual aspects of the technique are not new, but how it integrates them seemingly is.

When it comes to the research gap, it was not possible to find a closely related technique during the project's research. To try and find a closely related technique, I searched after related keywords (see *C. Appendix)* but was unable to find relevant results. In addition to this, articles that discussed 2fa, deception, and honeytokens did not mention them in a way that closely resembled the technique proposed in this paper. The chance always remains that the technique had already been employed but that I could not find evidence of this. Still, due to the relatively extensive amount of research conducted, this should not be the case. In addition to this, the implemented technique itself did – according to the analysis presented in *Chapter 5* – live up to expectations and was determined to be feasible. Since the implemented technique was (1)a new technique and (2)could be successfully implemented according to the definitions stated in *Chapter 3.7*,

the conclusion is that the research gap was filled.

When it comes to the project's relevance for other areas of cyber-security study than Masquerader Detection, the proposed technique can be relevant simply as a general 2fa technique. Even without the logging functionality and the implementation of the 2fa token as a deceptive honeytoken, the 2fa token could just be used individually. It would still provide an extra layer of defense while using even fewer hardware resources than it currently does. Besides this, however, the study is probably not of particular relevance for any other areas than masquerader detection. The technique is a niche technique and should be implemented together with techniques from other areas of cyber-security – since multiple researchers argued this integration of techniques to be highly relevant for the cyber-security field(see *Chapter 3.8*).

In regards to the potential contributions of the project, there does not exist any societal contribution. The completion of it has been limited exclusively to isolated testing environments. For the technique to be considered contributing to society, the tool should be further automated, tested on real-world servers, and in such a complete state that it could be provided to the public. If these aspects were fulfilled, then it would provide value to society.

Neither should this project be considered to contribute value to the industrial complex. No testing has occurred in the real world, and no feedback has been provided from, for example, it-admins working within the field. However, since phishing attacks are one of the most common ways unauthorized entities gain access to a system and thereby can masquerade as the authorized user, future research should contribute value to small, medium, and large companies.

However, a contribution to scientific research does exist. The core objective of this project was to develop a new technique for protecting a file system from a masquerader. Such a technique has been developed and tested. Additionally, according to the analysis of user-testing, checklist results, and – most notably – a cyber-security expert, the technique has been proven to be a feasible technique that should be further developed. Due to these factors, the project does contribute to scientific research.

8 – Future Research

During this chapter, details on four aspects for future research will be presented. This will contextualize how the technique could be developed further to improve current weaknesses. The reasons for the recommended future research will also be motivated.

One of the most important aspects of this project is that the testing only occurred in an isolated testing environment. The technique would have to be tested in a real-world situation to gather relevant information regarding how the technique would behave in a real-world environment. Future research should therefore include the migration of the tool into a real-world system, where it can run for a prolonged amount of time with a larger number of users interacting with it (these users including for example IT-admins responsible for the computer system's security). This will allow for more extensive validation of the tool.

Additional user testing scenarios should therefore be employed . This would evaluate during which of them that the technique is more/less feasible. During this project, the scenario was only computer-science students testing the tool within an isolated environment. Interesting future scenarios could, for example, be how a smaller company interacts with the tool. Or one could set up a private server on the internet and leave it open for intruders to see if/how they can break the technique. This would lead to results from a real-world environment instead of only a controlled and isolated environment.

The 2fa honeytoken should also be implemented for a virtual machine instead of for a single folder. Contemporary computer systems generally implement virtual machines (VMs) and containers due to many different reasons. A VM, for example, often enhances protection. So instead of the user logging in to the server and being provided with permissions for the protected folder when accessing the 2fa honeytoken, the following could be implemented: The user logs in via SSH but accesses a VM. The user is only let out of the VM when they have accessed the 2fa honeytoken. They are then provided with access to the real system (of course, the real system could also be a VM/Container).

The misinformation aspect of the tool is one of the most important subjects to research further, as evidenced by the many questions that the cyber-security expert had regarding it (see *Chapter 5.3)*. Results and analysis

on how to implement a complete misinformation environment that would fool intruders in a real-world scenario must be completed before the technique can be considered feasible/not feasible for real-world usage. This should include more extensive background research on the topic of misinformation and testing on how a comprehensive misinformation environment would affect, for example, resource usage and general usability. This is important to evaluate to determine how it affects users' interaction with the technique (perhaps the misinformation environment would confuse testers). One notable concern regarding this is how different user's file systems should be set up for the 2fa honeytoken to remain elusive to also an insider masquerader. Should users themselves be allowed to decide where the token is hidden? If so, does this lead to users choosing the same – or similar – locations, thereby undermining the technique's potential protection? As can be seen, there are many questions to answer in regards to this. Therefore, how the tool can be more deeply integrated with misinformation is considered one of the most important topics for future research.

References

[1] L. Lazarovitz, “Deconstructing the solarwinds breach” Computer fraud & security, 2021-06, Vol.2021 (6), p.17-19. [Online]. Available: https://www-sciencedirect-com.proxy.lnu.se/science/article/pii/S1361372321000658. [Accessed Feb. 13, 2021]

[2] P. Sandgren, ”Säpo varnar för intensifierade cyberattacker mot industrin“ Teknikföretagen, 2020. [Online]. Available: https://www.teknikforetagen.se/nyhetscenter/nyheter/2020/sapo-varnar-for-intensifierade-cyberattackr-mot-industrin/. [Accessed Feb. 13, 2021]

[3] V. Papaspirou, L. Maglaras, M. A. Ferrag, I. Kantzavelou, H. Janicke and C. Douligeris, “A novel Two-Factor HoneyToken Authentication Mechanism” Cornell University, 2021. [Online]. Available: https://arxiv.org/abs/2012.08782. [Accessed: Feb. 13, 2021].

[4] I. Homoliak, F. Toffalini, J. Guarzino, Y. Elovici and M. Ochoa, “Insight Into Insiders and IT: A Survey of Insider Threat Taxonomies, Analysis, Modeling, and Countermeasures” ACM Computing Survey, vol 52, no 2, pp. 56 30-40, March 2019. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3303771. [Accessed Feb. 13, 2021]

[5] H. Peng and W. Wang, “Detecting Masqueraders by Profiling User Behaviors” International Conference on Instrumentation and Measurement, Computer, Communication and Control, 19-21 July 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9045384/. [Accessed Feb. 13, 2021]

[6] A. Kellett, “Trends and Future Directions in Data Security – Vormetric Insider Threat Report” Vormetric Data Security, 2015. [Online]. Available: https://www.bankinfosecurity.com/whitepapers/2015-vormetric-insider-threat-report-trends-future-directions-in-w-1600. [Accessed Feb. 13, 2021]

[7] M. Schonlau, W. DuMouchel, W. Ju, A. Karr, M. Theus and Y. Vardi, “Computer Intrusion: Detecting Masqueraders” Statistical Science, vol 16, no 1, pp. 58-74, 2001. [Online]. Available: https://www.jstor.org/stable/2676780. [Accessed Feb. 13, 2021]

[8] S. J. Stolfo, M. B. Salem and A. D. Keromytis, “Fog Computing: Mitigating Insider Data Theft Attacks in the Cloud” in 2012 IEEE Symposium on Security and Privacy Workshops, San Francisco, CA, USA,

24-25 May 2012. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6227695. [Accessed: Feb. 13, 2021].

[9] A. Ahmad, J. Webb, K. Desouza and J. Boorman, "Strategically-motivated advanced persistent threat: Definition, process, tactics and a disinformation model of counterattack" Computers & Security, vol 86, pp. 402-418, September 2019. [Online]. Available: 57 https://www.sciencedirect.com/science/article/pii/S0167404818310988. [Accessed Feb. 13, 2021]

[10] R. Grimes, "The Many Ways to Hack 2FA" Network Security, vol 2019, no 9, pp. 8-13, September 2019. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1353485819301072. [Accessed Feb. 13, 2021]

[11] J. Castro and J. Mylopoulos, "The Feasibility Study", Information Systems Analysis and Design, 2002. [Online]. Available: http://www.cs.toronto.edu/~jm/340S/PDF2/Feasibility.pdf [Accessed Feb. 13, 2021]

[12] K. Peffers, T. Tuunanen, M. Rothenberg and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research" Journal of Management Information Systems, vol 24, no 3, pp. 45-77, 2007. [Online]. Available: https://www.tandfonline.com/doi/abs/10.2753/MIS0742-1222240302. [Accessed Feb. 13, 2021]

[13] Linnaeus University, "Course Room for Degree Projects" 2015.[Online]. Available: https://coursepress.lnu.se/subject/thesis-projects/validity/ [Accessed Feb. 13, 2021]

[14] M. Shaw, "What Makes Good Research in Software Engineering?" International Journal on Software Tools for Technology Transfer, vol 4, no 1, pp. 1-7, October 2002. [Online]. Available: https://link.springer.com/article/10.1007/s10009-002-0083-4. [Accessed Feb. 13, 2021]

[15] F. P. Brooks, "Grasping Reality Through Illusion – Interactive Graphics Serving Science" Human Factors in Computer Systems Conference (CHI'88) pp. 1-11, 1988. [Online]. Available: https://dl.acm.org/doi/10.1145/57167.57168. [Accessed Feb. 13, 2021]

[16] Linnaeus University, "Course Room for Degree Projects" 2015.[Online].

Available: https://coursepress.lnu.se/subject/thesis-projects/reliability/ [Accessed Feb. 13, 2021]

[17] D. Fraunholz, S. Anton and C. Lipps, “Demystifying Deception Technology: A Survey” University of Kaiserslautern, 2018. [Online]. Available: https://arxiv.org/abs/1804.06196. [Accessed Feb. 13, 2021]

[18] M. Leaf, “What is Formal Analysis?” Cybernetics and Systems, vol 35, pp. 2-3, March 2004. [Online]. Available: https://www.researchgate.net/publication/263335247_WHAT_IS_FORMAL_ANALYSIS [Accessed Feb. 13, 2021]

[19] PennState University, “Empirical Research in the Social Sciences and Education” 2021. [Online]. Available: https://guides.libraries.psu.edu/emp [Accessed Feb. 13, 2021]

[20] C. Faveri, A. Moreira, E. Souza, “Deception planning models for cyber security” Universidade NOVA de Lisboa, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8000014/. [Accessed Feb. 13, 2021]

[21] L. Spitzner, “Honeypots: Catching the insider threat,” Annual Computer Security Applications Conference, 8-12 December 2003. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1254322. [Accessed Feb. 13, 2021]

[22] A. de Barros, “Dlp and honeytokens,” 2007. [Online]. Available: http://blog.securitybalance.com/20070801archive.html. [Accessed Feb. 13, 2021]

[23] S. Srinivasa, E. Vasilomanolakis and J. Pedersen, “Towards systematic honeytoken fingerprinting” Association for Computing Machinery, Vol 4, No 7, 2020. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3433174.3433599. [Accessed Feb. 13, 2021]

[24] Thinkst Applied Research, “Canarytokens” 2021. [Online]. Available: https://github.com/thinkst/canarytokens. [Accessed Feb. 13, 2021]

[25] K. Adel, “HoneyLambda” 2018. [Online]. Available: https://github.com/0x4D31/honeyLambda. [Accessed Feb. 13, 2021]

[26] K. Adel, “Honeybits” 2019. [Online]. Available: https://github.com/0x4D31/honeybits. [Accessed Feb. 13, 2021]

[27] M. Bercovitch, M. Renford, L. Hasson, A. Shabtai, L. Rokach, and Y.Elovici, “HoneyGen: An automated honeytokens generator” International Conference on Intelligence and Security Informatics, 2011, Beijing China. [Online]. Available: https://ieeexplore.ieee.org/document/5984063 [Accessed Feb. 13, 2021]

[28] M. Lazarov, J. Onaolapo, and G. Stringhini, “Honey sheets: What happens to leaked google spreadsheets?” USENIX Workshop on Cyber Security, 2016, Texas USA. [Online]. Available: https://www.usenix.org/sites/default/files/conference/protected-files/cset16_slides_lazarov.pdf. [Accessed Feb. 13, 2021]

[29] L. George, P. Charalampos, L. Wilbanks, “Digital Deception: Cyber Fraud and Online Misinformation” IT Professional, vol 22, no 2, pp. 19-20, November 2020. [Online]. Available: http://gala.gre.ac.uk/id/eprint/28113/. [Accessed Feb. 13, 2021]

[30] A. Tunggal, “What is Access Control? The essential cybersecurity practice?”, 2021. [Online]. Available: https://www.upguard.com/blog/access-control [Accessed Aug. 22, 2021]

[31] P. Small, “Defense in Depth: An Impractical Strategy for a Cyber World” SANS Insitute, 2011. [Online]. Available: https://www.sans.org/reading-room/whitepapers/warfare/defense-depth-impractical-strategy-cyber-world-33896. [Accessed Feb. 13, 2021]

[32] O. Zohar, A. Barbalat, and R. Elhara, “Applying deception mechanisms for detecting for sophisticated cyber attacks: A research paper by topspin security.” 2016. [Online]. Available: https://www.helpnetsecurity.com/2016/10/12/deception-mechanisms-detecting-attacks/. [Accessed Feb. 13, 2021]

[33] D. Fraunholz, D. Krohmer, F. Pohl and H. Schotten, “On The Detection and Handling of Security Incidents and Perimeter Breaches - A Modular and Flexible Honeytoken based Framework” German Research Center for Artificial Intelligence, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8328709/. [Accessed Feb. 13, 2021]

[34] X. Han, N. Kheir and D. Balzarotti, “Deception Techniques in Computer Security: A Research Perspective” ACM Computer Survey, Vol 51, No 4, Article 80, 2018. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3214305. [Accessed Feb. 13, 2021]

[35] R. Love, “Kernel Korner - Intro to inotify”, 2005. [Online]. Available: https://www.linuxjournal.com/article/8478. [Accessed Feb. 13, 2021]

A. Appendix – Interviews

Observe that the non-structured parts of the interviews are not included. Only the structured predetermined questions asked.

Tester D:

Do you think this might be a relevant(feasible) idea to continue working on?

- Yes, absolutely. I have not heard about this idea before but it seems like a good idea.

Do you have any suggestions for improving the technique?

- I don't know if the tool is only meant for companies, but I think it would be perfect for individuals also when it comes to files you want to keep safe. 2fa can be irritating since they are so often things that exist externally that you always have to keep track of, but since this exists within the system it makes things easier I think. But I don't really know about any improvements. Maybe that you have most documents available and only hide the most important ones, so that the filesystem looks as real as possible.

I think it would be interesting if the tool could be connected to a certain hardware, so that the hardware itself is necessary for accessing the 2fa honeytoken correctly.

What are, in your opinion, the weaknesses/strengths of it?

- I think that depends on whether one accesses the token within the timelimit or not. The time limit is good, but if the user simply forgets then it becomes a waste of time for the admin. And if the token is still unlocked, then maybe the admin don't have time with the warning and the intruder can look through the system until it finds the 2fa honeytoken. Then maybe the token can be locked after 30 seconds, and then unlocked again after login.

Was it irritating to use?

- No, I only had to use cat in order to open the token.

Where would you want the 2fa Honeytoken to be hidden?

- Maybe in some folder where it really does not belong, for example /bin. Or in some folder that is not so important for a hacker. This also depends on the scenario. A hacker might look for different things in a companies and individuals computers. For an company's computer I would place the token in a folder with logo pictures. Try to make the folder as non-relevant as possible.

How would you design the token if it was up to you?

- I don't know.

Do you have any general thoughts in regards to this technique?

- Only what we talked about with the token disappearing if not accessed within the time limit. This can be a problem. So I think a better solution would be to lock the token and then unlock it after login. Also, if I would use this tool I would put the file among all of my savefiles for Sims!

Tester J:

Do you think this might be a relevant(feasible) idea to continue working on?

Yes. I don't have a really in-depth understanding of it, but I understood the core idea, so I think yes.

Do you have any suggestions for improving the technique?

Don't know.

What are, in your opinion, the weaknesses/strengths of it?

The strength is that you access the token afterwards instead of before.

Was it irritating to use?

No really.

Where would you want the 2fa Honeytoken to be hidden?

I would put it in the pictures folder.

How would you design the token if it was up to you?

Don't know.

Do you have any general thoughts in regards to this technique?

Nope.

Tester L:

Do you think this might be a relevant(feasible) idea to continue working on?

I think it depends on how well hidden the token is. But if the token is well hidden, then yes.

Do you have any suggestions for improving the technique?

This token, I would like to have it located so that it is harder to access it than right now. Maybe password protect it? But perhaps this makes it easier to find it with a scan.

What are, in your opinion, the weaknesses/strengths of it?

It is an extra protection. Not sure what other similar alternatives there are, but in any case this is an additional protection.

Was it irritating to use?

No.

Where would you want the 2fa Honeytoken to be hidden?

In some folder with systemfiles, so that it is hard to distinguish it as being a 2fa file.

How would you design the token if it was up to you?

Not really any thoughts besides adding the password.

Do you have any general thoughts in regards to this technique?

No, I think I have said more or less everything.

Tester R:

Do you think this might be a relevant(feasible) idea to continue working on?

Yes. If you can develop in in a good way I might be interested in using it for my own project servers. I think it is a very good idea since a hacker doesn't know that this protection is here.

Do you have any suggestions for improving the technique?

Maybe you could make it so that the whole filesystem is fake? So that the hacker thinks they have access to the whole system.

What are, in your opinion, the weaknesses/strengths of it?

If the token is located so that it is to easy to find it with a scan.

Was it irritating to use?

The file system is so limited right now so it is hard to tell. But right now it is not irritating to use it.

Where would you want the 2fa Honeytoken to be hidden?

Somewhere deep inside the system. Maybe in the /etc folder. But the question of access is also important.

How would you design the token if it was up to you?

Can you add a password to it also? Maybe a random password? Anyway so that it is not possible to bruteforce it within 30 seconds. I think that like now, without a fileending is better, it makes it harder to scan for it.

Do you have any general thoughts in regards to this technique?

I think that if you could make it so that the place where people login is a jumpserver, and then when the 2fa is accessed, the user can access the real ssh key. Anyway, I have not come across this way of using honeytokens before. But I think if the user knows that there are honeytokens like this, then that might be a problem. Is it not possible to see the honeytoken location via the .bash_history? Maybe delete this file after the user logs out.

B. Appendix – Information for Testers

The total number of testers of this project's implementation are 4. You will access the tool that implements the proposed technique by SSHing to a dedicated server. On this server there exists two user accounts:

- Admin(aw222zr) – from where the tool is run.
- Tester(testare1) – the user which the tool is run against.

The account you will access is obviously the Tester account. You will share the tester account and SSH key with the other testers. Testing will occur live during an explanation of the tool, where you can give feedback etc.

When you log in, you will access a "fake" file-environment. In this fake file-environment there also exists a 2fa honeytoken. In order to access the "real" file-environment, you need to activate this token(can be done by for example running cat against it, or opening it with Vim). If the token is not accessed within 30 seconds, an alert will be sent to the admin, and information about the user, date, ip-address etc will be logged. If this alert is raised, accessing the token will no longer lead to the real file-environment being unlocked.

The location of the tester user's home-folder is */home/testare1*, and this folder is populated with data intended to represent both fake and real information. The files to be protected by the 2fa token are only the most important files – files that for example contain business secrets. The rest of the file-environment should be as real as possible, in order to not raise suspicion.

The location of the tester user's real folder is */usr/local/testare1*. This folder has root as default owner, and is locked down with 000 permissions until the 2fa token is accessed. When the user accesses the 2fa token, permissions are set to 700 and the owner is set to testare1.

I will save the results of the testing in files with coded names. I will remember which tester has what code. This coding of each tester's name will ensure the protection of each tester's data.

C. Appendix – Literature Review Searches

The first section is the keywork, the second the number of search results. The searches were made via Google Scholar.

- 2fa - 32 200 Results
- Honeytoken - 883 Results
- Honeytoken misinformation - 96 Results
- Misinformation - 299 900 Results
- Cyber security deception - 35 100 Results
- Inotify - 1110 Results
- 2fa honeytoken - 14 Results
- Misinformation defense - 64 000 Results
- Solarwinds breach - 670 Results
- Honeytoken authentication - 395 Results
- Masquerader security - 2170 Results
- Cyber insider - 54 600 Results
- Intrusion detection masquerader - 12 600 Results
- Disinformation security - 51 000 Results
- Feasibility study computer science - 2 930 000 Results
- Defense in depth cyber - 63 500 Results
- Layered defense cyber security - 30 900 Results
- Access control cyber security - 595 000 Results