Development and Assessment of User Interface for Security-Critical Systems

Truong Hoang

2015

Master's Thesis

Department of Design Sciences
Lund University

Abstract

In this master's thesis a user interface for a security application was investigated. The thesis work was done at a security company in Sweden. The goal was to find current problems with the user interface and to develop a new prototype with improvements. A user-centered design approach was employed to achieve this. Another goal was looked at how this method works for security-critical systems and how usability affects security.

By conducting a usability test, usability errors were found that affected information security. There were also other problems that arose when having to deal with trying to use a user-centered design approach when working for a security company.

Keywords: usability, user interface design, security-critical systems, information security

Dedication and acknowledgements

First and foremost, thanks to my thesis mentor, Joakim Eriksson, for his help and advice during my thesis work.

I also want to thank my mentors at the company I worked at, though I can not mention them by name. Thank you for allowing me to work at your company and providing me with everything I needed to do my thesis.

Finally, thanks to my family for always supporting me and their unconditional love which allows me to be me and made all this possible.

Table of Contents

List of Tables

List of Figures

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Background

For users of high-security software it is critical for them to be able to configure their product properly and with as few errors as possible. Poorly designed interfaces can cause potential disasters. An example would be if top secret information is leaked or the *wrong* information is sent.

Usability and security are often considered opposite of each other. Finding the right balance between having a secure system and having a system users can actually use is crucial for a security-critical system. User errors can be as dangerous as security issues in a software program.

A cyber-security company in Sweden, that wish to stay anonymous, which provides VPN services to different organizations around the world are currently looking to improve on the user interfaces for their products. They will be referred to as *CompanyX* from here on out.

Designing user-friendly interfaces requires knowledge in how the human brain works, cognitive behavior, and widely accepted design principles. We need to understand the user, what they want to accomplish and how we can design to meet those requirements.

Human-computer interaction (HCI) is about looking at different methods and design processes for designing interfaces for humans. It also looks at what tools that should be used to implement those designs. HCISec (Human-computer interaction (security)) is HCI but from the perspective of usability for security applications for the end-user.

In this master's thesis, I examined an application made by CompanyX and the application's existing user interface. The goal was to find current problems with the user interface and to develop a new prototype which they could use. I used tools and design principles and also analyzed how well they worked for the company and their products. The goal was to find a design process that works for the company and security-critical

systems and to improve on their current products.

Some of the main topics of this master's thesis are *"Security vs Usability"*, *"Usability Testing of Security-Critical Systems"*, *"Heuristic Evaluation of Security-Critical Systems"* and *"Design Processes for Usable Security-Critical Systems"*.

1.2 Existing Product

The existing product that I looked at in this master's thesis is a VPN (Virtual Private Network) encryption product that enable organizations to send classified information between two protected networks over an unsecure network like the Internet. It works by creating a secure tunnel between VPN devices which protect protected networks.

These state-of-the-art VPN encryptors meet high-level security requirements and are used by multiple organizations for VPN security on IP-based networks.

To use these network encryptors, the company has developed a configuration application for the VPN devices. The configuration application will be referred to as *ConfApp* in this report. It is used by various network and system administrators to set up tunnels between devices and to configure them with various options. In the application you set up configurations. The configurations have information about the various VPN devices, the network information, and the tunnel parameters. These configurations are then exported and installed on the physical devices.

CHAPTER 2

THEORETICAL BACKGROUND

This section will describe the theoretical background for this master's thesis. This will be the foundation for the rest of the work. It will attempt to connect different parts of the *Information Security* and *Human-Computer Interaction* fields to create something that *works*.

There are several popular methods, processes, and design principles that are used to help make a software usable. They do not, however, guarantee usability but should be seen as tools. Use them wisely and they will be of tremendous help but as with any tool, use it incorrectly and they will not do anything.

Some of the more popular ways to design usable interfaces are by involving the user early in the design process, using heuristics and similar techniques to analyze interfaces and to design early prototypes. Early prototypes minimize costs and reduce commitment to one specific user interface that in the end might not even work.

The issues arise when we try to align security and usability. These tools need to be modified slightly and we need to apply our knowledge about information security, security mechanisms, and risk analysis to be able to understand how we can make secure *and* usable software.

2.1 Usability

2.1.1 Definition

There are many definitions of the term *usability* but in this thesis I will use the one provided by the ISO 9241-11 standard.

> The effectiveness, efficiency and satisfaction with which specified users achieve specified goals in particular environments.
>
> *ISO 9241-11*

While there are many definitions, the main point is that it is the user that is in focus and what he wants to accomplish. From usability we get *User-Centered Design* where you involve the user early in the design process. The term was first introduced by Don Norman in his book *Design of Everyday Things* [1]. For a security company with high demands from its customers to keep things confidential, it is a bit more difficult to involve the user. However, *User-Centered Design* does not necessarily need to involve the user early on but instead we can just have the user in mind in all stages of the design process and how the system will work for them. We can do this by trying to understand the end-users' needs and wants. We also want to try to understand how users will use our product.

2.2 Heuristic Evaluation

One way to evaluate how well designed a user interface is from a usability perspective is to do a *Heuristic Evaluation*. It involves going through a design using a specific set of metrics and evaluating the design on how well it does with these metrics.

There is a similar technique that is called *Cognitive Walkthrough*, the difference being that Cognitive Walkthrough focuses on using tasks to go through a user interface. Nielsen's heuristics are among the most widely used ones when it comes to Heuristic Evaluation. They are defined as follows [2]:

Visibility of system status: The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Match between system and the real world: The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

User control and freedom: Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Consistency and standards: Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention: Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

Recognition rather than recall: Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Flexibility and efficiency of use: Accelerators-unseen by the novice user-may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Aesthetic and minimalist design: Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Help users recognize, diagnose, and recover from errors: Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Help and documentation: Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

The second item in the list is something that should definitely be kept in mind when designing usable security software. Computer Security and Information Security tend to use terms that are not very well known and can cause confusion for the end-user if he is not well-versed in security terminology. This, of course, leads back to one of the first things noted in this paper; focus on the user and what he or she wants to accomplish.

The usability problems found using heuristic evaluation could all be motivated either by the heuristics listed above or known usability principles. What Heuristic Evaluation *does not* do is according to Nielsen: "Heuristic evaluation does not provide a systematic way to generate fixes to the usability problems or a way to assess the probable quality of any redesigns. However, because heuristic evaluation aims at explaining each observed usability problem with reference to established usability principles, it will often be fairly easy to generate a revised design according to the guidelines provided by the violated principle for good interactive systems. Also, many usability problems have fairly obvious fixes as soon as they have been identified" [3].

2.3 Similarity and Contrast

> Similarity and contrast, connection and separation, grouped and ungrouped are all ways to describe the varying sameness and difference between elements. Based on the information they carry, we'll want some elements to look similar, to indicate that they are related in some way. We'll also want to show that some elements are different and belong to different groups.
>
> *Steven Bradley* [4]

Contrast is a way to Contrast and similarity are some of the most powerful tools when designing user interfaces. These two concepts will be used throughout our analysis of the ConfApp application.

2.4 Information Security

CompanyX works with information security and the majority of their products are designed to enforce information security. *Confidentiality, Integrity, and Availability* are the key terms in information security. ConfApp is a tool to enforce information security. The keywords for information security, also known as the *CIA Triad*, are described as follows [5]:

Confidentiality: Assurance that information is shared only among authorized persons or organizations. Breaches of Confidentiality can occur when data is not handled in a manner adequate to safeguard the confidentiality of the information concerned. Such disclosure can take place by word of mouth, by printing, copying, e-mailing or creating documents and other data etc. The classification of the information should determine is confidentiality and hence the appropriate safeguards.

Integrity: Assurance that the information is authentic and complete. Ensuring that information can be relied upon to be sufficiently accurate for its purpose. The term Integrity is used frequently when considering Information Security as it is [sic] represents one of the primary indicators of security (or lack of it). The integrity of data is not only whether the data is 'correct', but whether it can be trusted and relied upon.

Availability: Assurance that the systems responsible for delivering, storing and processing information are accessible when needed, by those who need them. The concept of availability means that the information, the computing systems used to process the information, and the security controls used to protect the information are all available and functioning correctly when the information is needed

2.5 AEGIS Process

Appropriate and Effective Guidance for Information Security (AEGIS) is a software development process for secure and usable systems [6]. It was examined by Flechais in his Ph.D. thesis for designing secure and usable software [7, p. 57-63] . Much of the research in usability for security-critical software have mostly been focused on analysis of user interfaces of security-critical software and the issues with these. Not as much research has been put into developing and suggesting design processes that prevent these problems from occurring. Much of the difficulty in this is that companies and developers of security-critical systems have strict rules about their software and who gets access to their software and to test during their development process.

The AEGIS process is based on involving stakeholders early on in the process and doing risk analysis early on in the design process of software engineering.

CHAPTER 3

INITIAL USABILITY TEST

Usability testing is one of the most popular and best methods for finding user errors. It has been advocated by several usability experts [8][9][10]. I used usability testing in my master's thesis work as one of the main components for finding usability problems in the configuration application. The difficulty in doing usability testing or any kind of usability research for a security company is that they are not that inclined to share information about what they do, who their customers are, and what they are working on. This can be quite a problem when doing usability testing as you want to understand the customers, their problems, and goals.

3.1 Purpose of Initial Usability Test

My main goal for this usability test was finding usability problems that affected information security. I wanted to see what errors users made that could affect *confidentiality, integrity, and availability*. Another goal was to find general usability problems that made it difficult for users to do what they wanted to do in the ConfApp application.

3.2 Prestudy

To be able to conduct and plan for a usability test I had to learn more about CompanyX and the ConfApp application. It was imperative that I understood the product and what it was used for to be able to plan a test. If you can not understand in what context the product is used and how it is used then it is nearly impossible to create user tasks to simulate a real-world scenario in the usability testing sessions [11].

3.2.1 Domain Specific Application and Usability Testing

The complexity of the ConfApp application makes it hard for a user to understand it if he is not well versed in the internet protocol and network domain. Domain knowledge is something many people who work in the usability field struggle with and domain specific products that need to be tested are generally more difficult for usability test conductors to test [11]. Most of what is considered difficult is for the usability test conductors to understand the domain and to learn it.

Another issue that arises is that the end-users for security-critical systems are usually not available for usability tests. This causes problems for the usability test conductor because he has to find ways to try to emulate the real-world user. Additionally, the usability expert might not know who the end-user is but instead has to use his imagination and knowledge to come up with a persona that fits with what he thinks is the end-user.

3.3 Planning a Usability Test

3.3.1 Recruiting Participants

It was initially planned that I would use around 10-15 testers, but it was changed to five as we realized that finding that many testers would be difficult as I was only allowed to recruit inside CompanyX. According to Nielsen, you only need around five persons in a usability test to find 80% of the problems [12].

3.3.1.1 Prerequisites

The ConfApp application was not developed for any user but for system administrators who have knowledge in how networks are work. This needed to be reflected in my usability test which meant I had to do a screening at the initial stages of my recruitment. The company also runs a course where they teach their customers how to use their application. This meant for me that I also had to find people who had a basic knowledge of ConfApp. They also could not be experts.

I sent out an e-mail (see Appendix D) to the entire company to recruit test users. I outlined the prerequisites and also how and when the test was going to be conducted. The time schedule was quite flexible to accommodate for the issue with the staff at the company being very busy. Many of their deadlines were near the end of spring so this caused some clashes with my test dates.

3.3.2 How and What to Test

I chose the ordinary way to test by letting the user test the application by trying to complete user tasks. While they were doing this, I would observe them and see what they did. This allowed me to see any potential mistakes/problems they encountered during

their test. At the end of the test session I asked them some questions about the test and how they felt about the application.

3.3.3 Test Location and Setup

The tests were conducted in one of CompanyX's conference rooms. It was a basic layout where we had a desk with a laptop. The test coordinator sat on the right side, next to the participant.

3.3.4 Video and Audio recording

The usability test was conducted by one person and it is difficult to observe, take notes and talk with the user all at the same time. You can easily miss what the user is doing if you are taking notes. You can also forget things if you do not take notes. To remedy this, I decided I would record my test sessions to make sure I did not miss any user action. A screen recorder called *Open Broadcaster Software* [13] was used to record the screen and audio from the built-in microphone on the computer.

3.3.5 Test Metrics

These are some of the metrics that were used in the usability test. They were taken from usability.gov [14].

Successful Task Completion

Each scenario requires the participant to obtain specific data that would be used in a typical task. The scenario is successfully completed when the participant indicated they had found the answer or completed the task goal.

Critical Errors

Critical errors are deviations at completion from the targets of the scenario. For example, reporting the wrong data value due to the participant's workflow. Essentially the participant will not be able to finish the task. Participant may or may not be aware that the task goal is incorrect or incomplete.

Non-Critical Errors

Non-critical errors are errors that are recovered by the participant and do not result in the participant's ability to successfully complete the task. These errors result in the task being completed less efficiently. For example, exploratory behaviors such as opening the wrong navigation menu item or using a control incorrectly are non-critical errors.

Error-Free Rate

Error-free rate is the percentage of test participants who complete the task without any errors (critical or non-critical errors).

Likes, Dislikes, and Recommendations

Participants provide what they liked most about the software, what they liked least about the software and recommendations for improving the software.

3.4 The Initial Usability Test

3.4.1 Participants

I found five people who I tested the product on. Most of them were familiar with the configuration application and some had even used the program in the past (not daily but at least a couple of times).

They all worked at the company and could be considered experienced computer users. They worked with computers on a daily basis. A majority of the test participants were either system developers or software testers. This put a high overall computer aptitude bar for the participants.

3.4.2 Test Session Introduction

The users were given an introduction verbally. The introduction was read from a pre-written paper to ensure that all users got consistent information. The users were briefed on what to expect from the test and were given an opportunity to come with any questions or concerns. They were encouraged to think out loud and were notified about that the test would be recorded.

The main purposes of the introduction were to make the user feel comfortable and understand what they were to do.

3.4.3 Prerequisite Training

To ensure that all participants had a basic knowledge of the system and functionality, I conducted a short prerequisite training before giving the participants the tasks. It involved explaining how a physical VPN system in our case typically looks like and how it works. I also showed a bit of the application, basic operations such as creating Groups, Nets and VPN devices and how this correlated to the real-world VPN system.

3.4.4 Scenario and Tasks

The goal was to create around five to ten tasks for the users to try complete in a one-hour session. I ended up creating six tasks (see Table 3.1 for scenario and user tasks and Appendix A for input data). The first two tasks were relatively simple. This was done to make the user more comfortable and letting him get a sense of how usability testing is done. Task 3 and 4 were the most difficult ones and the last two tasks were a bit easier to not discourage the participants if they failed at task 3 and 4.

The tasks were created by looking at the context of the application and how it was used by the real users. I was given a thorough walk-through by one of the company's employees who had been apart of the development of the application. By understanding the context and having used the program, I could create task scenarios that emulated real-world usage.

The task scenarios were given to the users for them to read themselves. By having the users reading the tasks themselves, they were not forced to remember what they are supposed to do. This removed the cognitive load and allowed them to focus on completing the task.

When I designed the tasks I had in mind how long they would take. I did not, however, measure how long it took for the user to finish a task. This was because the time to finish a task was not as crucial as doing a task correctly and without any security-critical errors.

3.4.4.1 Scenario

This was the scenario given to the test participants:

You are a system administrator for a large IT company and the company has several offices around the world. Two offices in Sweden, office A and B, have decided they want to share company information between each other in an easy and secure way. To do this your company has decided to use CompanyX's ConfApp product and its configuration application. The company has bought two VPN devices and placed one at each office location. Each office has a protected network where it's safe to send and receive data, but to send and receive data between the two offices you need to go over the internet. The VPNs are used to make this data transfer secure.

Table 3.1: Tasks

#	Description
1	Your job as a system administrator is to use the ConfApp configuration application to create configurations for the two VPN devices so they know how to communicate with each other. Set up a basic VPN configuration using shared key and a 256-bit encryption.
2	Your company has deemed it unnecessary to use a 256-bit encryption and asks you to change it to 128-bit.
3	Office A does not want all of their computers in their network to be able to communicate with office B. Make it so that computer 1 and 2 in office A cannot communicate with office B. All other computers in office A should still be able to communicate with office B.
4	You are afraid that if one of the VPN devices stops working that you will not have a connection between the two offices. Create backup VPNs for office A and B that take over if the main VPN devices that you set up previously fail.
5	Make sure your configuration meets all the security policies in the application that have been pre-defined by your company and save it so it can be installed on the VPN devices locally.
6	You go home for the day and come back the next day to see that one of the networks in your configuration application has been deleted. Find out when it was deleted and by whom.

3.4.4.2 Think Aloud

One of the best and most common ways to understand what a user is doing in a usability test is to ask him to use the *Think Aloud*-method which means having him verbalize his thoughts and what he is doing as he is trying to complete a task [10].

The downside of this technique though is that it makes the user much more aware of his actions and makes him more careful. This might cause a problem because sometimes the user might think more before doing a thing which he normally would just do. It is one of the main criticisms of this method, but I felt that the pros far outweighed the cons [10].

3.4.5 Results

In the results, I try to summarize the most significant results from the usability test from the goals that were laid out at the beginning of the chapter.

3.4.5.1 Successful Task Completion

Figure 3.1 shows the number of participants that successfully completed tasks 1-6.

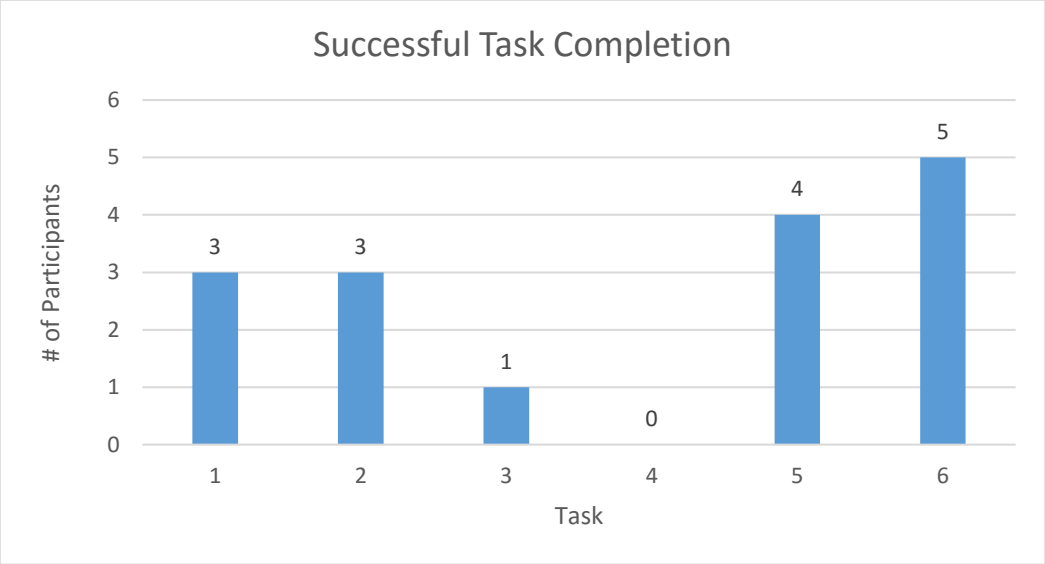

Figure 3.1: Amount of people who completed each task.

3.4.5.2 Security-Critical Errors

Two of the five test participants failed to configure the encryption length (hash length in the application). This could be considered a major problem if they did not realize what hash length meant. The main problem was finding where to set the encryption. Thankfully, the default was 256-bit encryption.

Four out of five participants were unable to set up an access list blocking specific computers from one protected network to communicate with the other. This could be seen as a security issue. Three of the four were able to create an access list but did not select it to be used in the tunnel properties. One participant, however, did realize that he missed that part and corrected it after when he had moved on to task 4.

All participants failed to set up a backup VPN (i.e., failover). They all created parallel VPN devices in the configuration application that ran in parallel with the main VPN device. Two participants realized this was not exactly a backup but failed to figure out how to use the failover setting even after being told that it was the failover option they should be looking for and after they had read the manual. Only one participant got as far as understanding that the IP address he set up previously on the main VPN device would become a virtual IP address. He, however, did not understand what virtual meant and did not understand he needed to add *two* new physical PLS (Plain Text Interface) and CIS (Cipher Text Interface) addresses for respective VPN device. PLS and CIS addresses follow the Red/black concept [15] where you carefully separate signals that carry classified information in plain text (red signals) from signals that carry encrypted information in cipher text (black signals). This task also took the longest time without any successful task completion.

3.4.5.3 Figuring Out How to Create a Tunnel

Three out of five participants did not understand how a P2P service worked. When asked to set up a basic VPN system they thought you were supposed to create a P2P service first without having created any nets or VPN devices. They all quickly recovered from this error when they saw that they had no peers to select from in the P2P wizard window.

One participant started with creating two groups, one for each net. This was due to an error by the test coordinator not giving him the proper prerequisite training as the participant had expressed his knowledge in the application and general reluctance for the need of a prerequisite training.

3.4.5.4 Exporting the VPN Device Configurations

One participant failed to export the VPN configuration so it could be installed on the VPN devices locally. He instead exported the system configuration XML file, which is not the configuration for the VPN but just shows changes done to the VPN system.

3.4.5.5 Finding Changes Made in the System

All participants were able to find the log where all changes to the system configuration were logged.

3.4.5.6 Error-Free Rate

None of the participants were able to complete any task with zero critical or non-critical errors.

3.4.5.7 Likes, Dislikes, and Recommendations

Generally, all participants found the program difficult to use. Some did justify it by saying that if they would use the application a couple of times they would have much less trouble finding things.

There were mixed reviews when it came to the user interface. Many liked the tree view as it gave them a good overall overview of the system. They did not comment on the icons. They thought there was too much information at some points that they felt no need of.

A couple of the participants suggested that hiding some options would be better and let more advanced users who want those options to be able to open them in an *Advanced*-feature. Information overload was one of the main complaints of the application. A user voiced his major dislike for the tab menu used in the VPN Edit Properties window. In particular the multi-level tab and how the tabs switched places depending on which tab you had as the active one. He said it confused him because he usually tries to remember where a tab is and uses that as a guide to finding it again.

The terminology in the application was confusing for all participants. They felt that some labels were badly worded and they did not understand what the labels meant. An example of this would be *hash length*, which they did not understand, or *Default set of tunnels* where the application did not explain what a default set of tunnels was.

The participants also found that the location of some menu options were unintuitive. All participants struggled to directly find the menus options they were looking for and had to do some exploration to find them. An example of this was where to change tunnel properties. A majority thought it was located under the P2P service they had created.

The majority complained about the flow of the program. They found it difficult to know what they should do next and instead had to think and figure it out themselves. There was a general positive response to the wizards used throughout the program though one user did remark on not liking that a Wizard automatically popped up to create a VPN device even though he had not asked the program to do this. One could argue more advanced users might find the wizards annoying.

The participants were asked if they thought the application was good at emphasizing important information and all said no. It was also from this question many suggested that maybe there was much information in the application that was not needed to be shown to the ordinary user

CHAPTER 4

LOW-FIDELITY PROTOTYPING

With the results obtained from the initial usability test I set out to improve the current software from CompanyX. Low-fidelity prototyping enabled me to come up with quick mockups that had a very crude look but were easily changed.

I used *Balsamiq Mockups* [16] for creating mockups as it was a tool I was familiar with and had all the necessary features I needed. See Appendix D for additional screenshots of the ConfApp application.

4.1 Design Problem One: User Workflow

4.1.1 Staging

A *staged* user interface is one that is intentionally designed to shepherd the user through a sequence of stages of system use, in order to increase user understanding and protect against errors [17].

One of the main challenges for the test users were that they did not know where to start and where to end. Logically, one would want to fix this problem by implementing *hard staging* [17], which is a stricter version of soft staging. This would allow the user to know what he should do next and reduce the chance of him making errors or getting lost. However, as this is a very domain specific application and I was dealing with expert users it can frustrate them if they do not have choices and have to follow a strict pattern.

For these reasons, I decided to stay with the *soft staging* [17] that is currently employed by the original application through the extensive use of wizards. What I did want to change, though, was to keep this consistent over the entire application.

As mentioned in chapter 3, I saw that many users failed to select an access list after they created one. This was because where you created the access lists was not where you selected to use them. By allowing the user to create a list where they select a list,

I would make it easy for them to follow a desirable progression, see figure 4.5. This fit perfectly with what soft staging is about.

Another area where users struggled was with the tunnel creation system. It was a huge disconnect going from the P2P Wizard to the tunnel configuration window. I decided it was advantageous to make the tunnel creation window into a wizard. This created the flow I wanted but still allowed users to select their configuration settings.

Consistency is key when designing interfaces and this is what I tried to achieve with these changes.

4.1.2 Wizards

I decided to keep wizards as a they were an excellent tool for making complex operations less intimidating. They also adhered to the staged user interface pattern. In my initial usability test, the users also seemed to be overall happy with using wizards. There were some problems with the original wizards that were used in the ConfApp application. I noticed in the usability test that the help text was rarely used. It is generally not advisable to put the help text to the left of the wizard under the main heading, see figure 4.1. Users often skip reading the help text and their eyes focus instantly on the main pane. In the original wizards this lead to the users not noticing the main heading because it was above the help text window.

A simple solution to fix this was moving the help text to the right. This allowed the users to instantly focus on what they wanted to see and ignore the help text if they wanted. Another problem that I noted was that the help texts were very extensive and complicated. Our test users noted this and said that perhaps the help texts should be shorter and more concise. I also added a graphical progress indicator (see figure 4.2) to let users more easily see where they are in the wizard. This is in line with visibility of system status from Nielsen's heuristics.

Another thing I did was to trim down the wizards, see figure 4.3 and 4.4. Many parameters that did not need to be configured were still shown in the original application, even though a user had chosen "Skip optional steps". A step does not necessarily involve skipping all optional parameters but I felt that users did not care about parameters that already had default values entered for them but instead only cared about the fields that *had* to entered.

In the ConfApp application you have to, at some steps in the wizard, enter a *netmask* (see fig 4.3), where you have an IPv4 address field and a netmask field. A netmask in computer science is a 32-bit number that divides an IP address into subnets. In the original application you can enter a subnet mask in three different forms:

1. *X.X.X.X*, e.g., *255.255.255.0*.

2. *X*, e.g., *24*.

3. In the IPv4 address field: *X.X.X.X*/*X*, e.g., *192.168.0.1*/*24*

The first two forms you enter your input directly in the netmask field. When you enter it in the second form it will automatically turn into the first form when you leave the text field. You can also enter a netmask directly in the IPv4 address field and the netmask field will be filled in automatically as you leave the IPv4 field. The original application itself does very little to let users know about these options and in my tests I noted that users did not know about this. A simple solution to this would be to add a small indicator next to the netmask field which you can hover over to learn about this or use a prompt text in the text field.

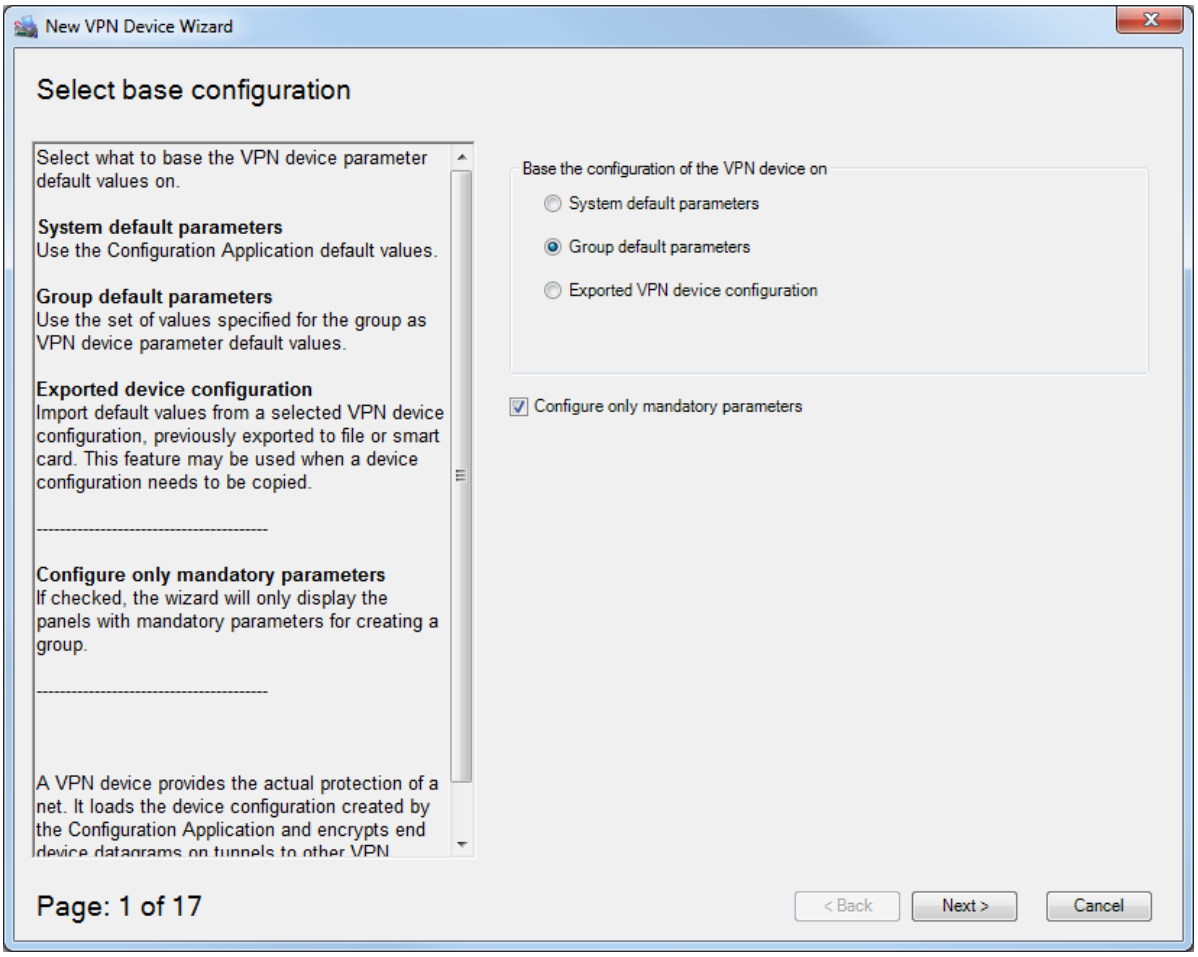

Figure 4.1: Original Wizard Step 1 Design

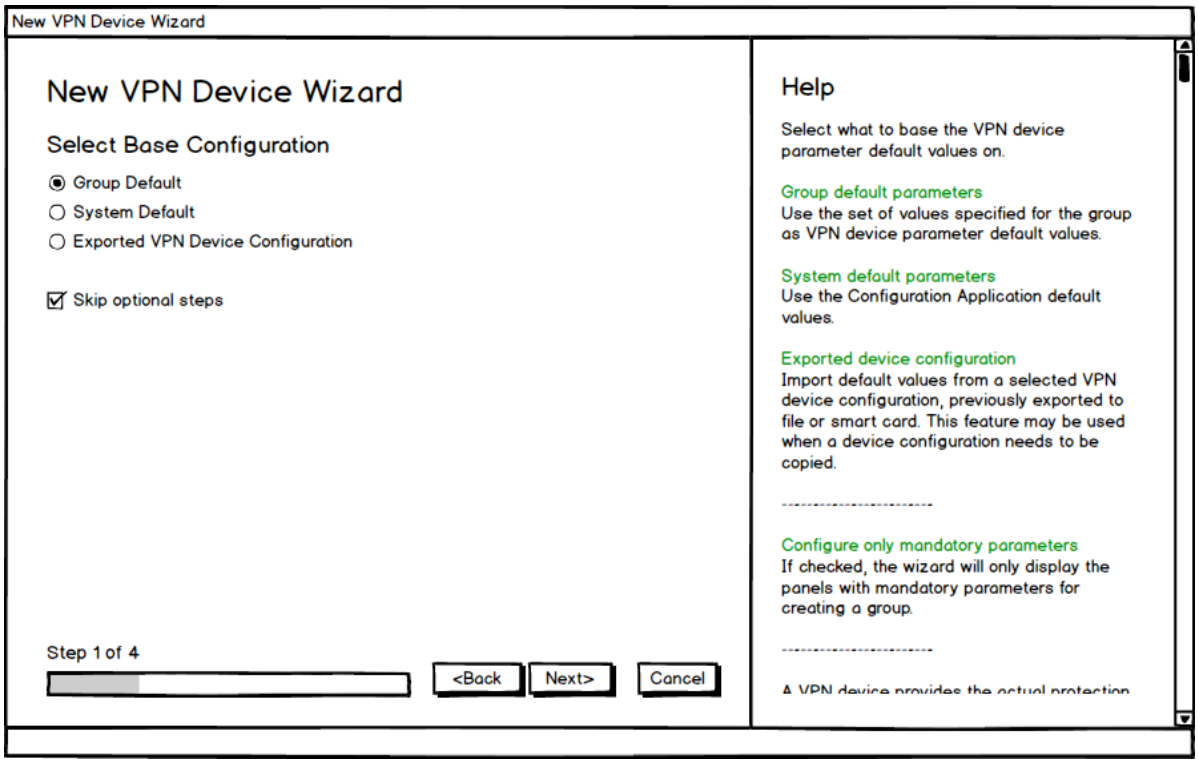

Figure 4.2: Low-Fidelity Wizard Step 1 Design

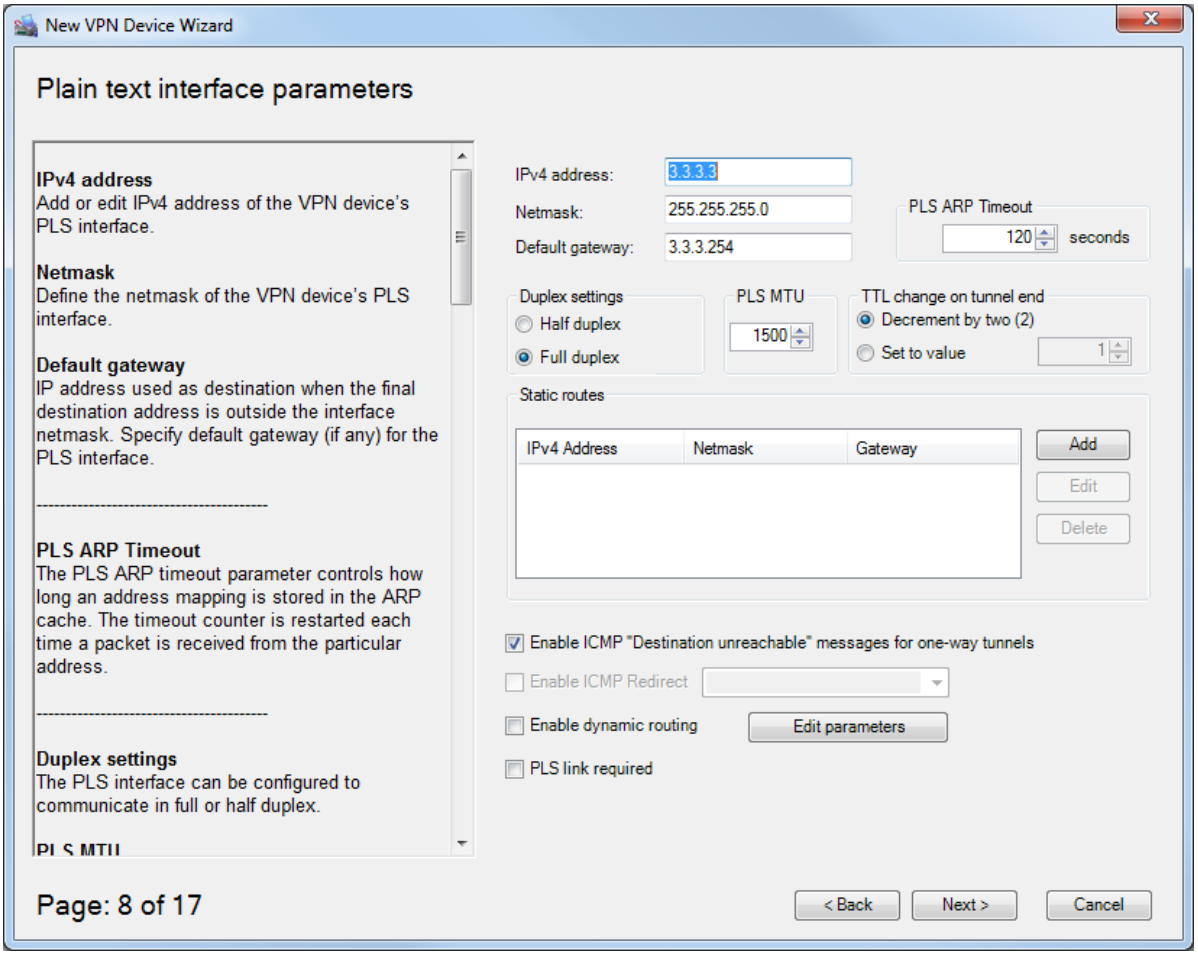

Figure 4.3: Original Wizard Step 3 Design

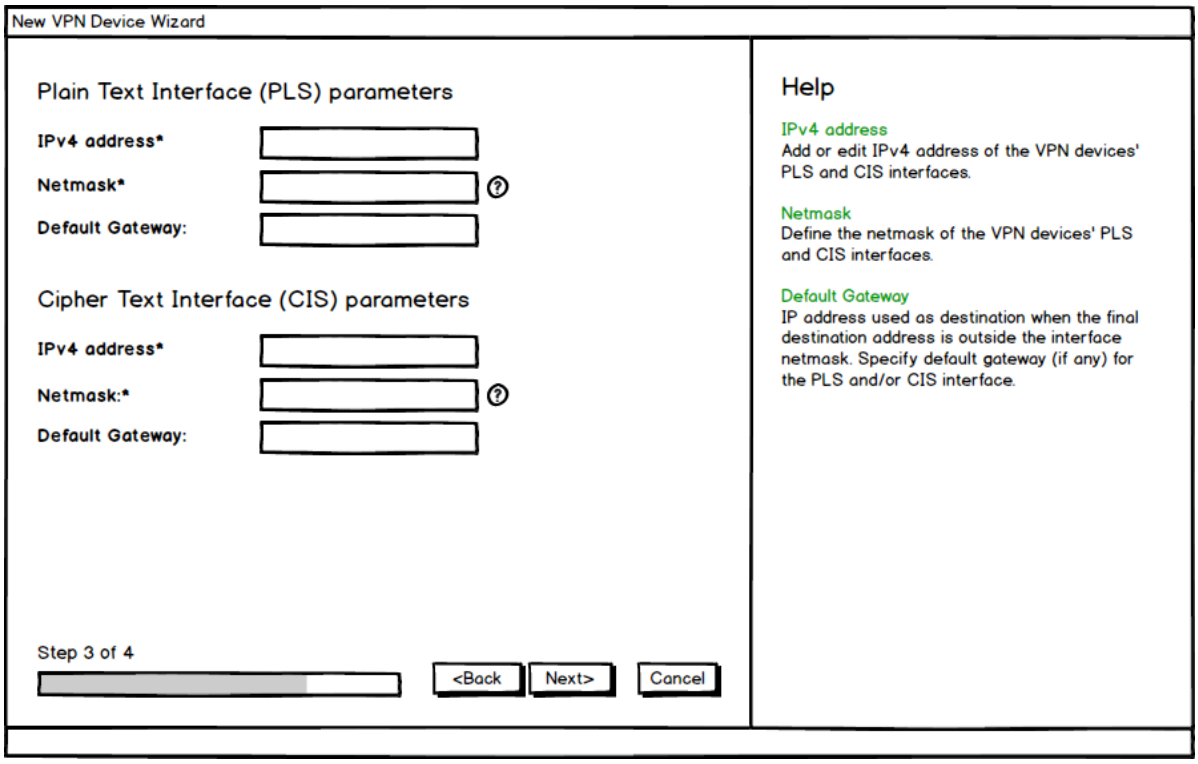

Figure 4.4: Low-Fidelity Wizard Design

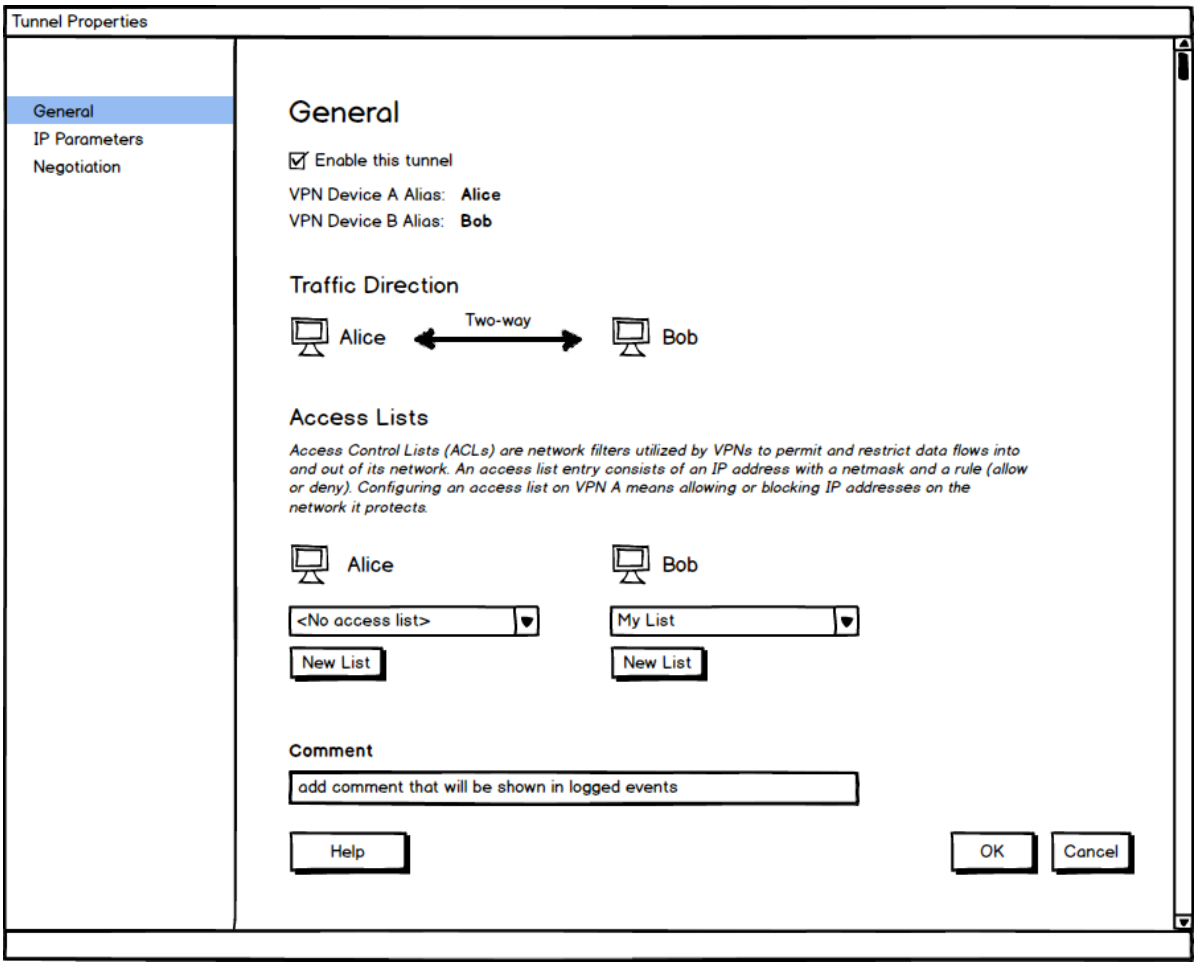

Figure 4.5: Tunnel Properties and Access List Design

4.2 Design Problem Two: Properties Windows

> Every time you release an apple over Sir Isaac Newton, it will drop on his head. That's good.
>
> *Jakob Nielsen*

Security applications are usually complicated and have many options or parameters. This is quite notable in the properties window for a VPN device in the ConfApp application. The properties window uses a multi-tab navigation method. I noted in my usability test that this caused confusion and users expressed their dislike for it.

The main problem with the current tab system (see figure 4.6) is that it creates disorientation. Users use familiarity and rely on consistency when navigating user interfaces [18][19]. What this tab system does is removing these two aspects. When you click on a tab on another row than the tab you currently have selected the entire tab row moves. This makes it impossible to memorize where tabs are by their position because they are constantly moving.

Ideally when having to deal with this many options one should think about if all are really necessary and if you can hide some of the options that are not used often. However, when you work with security applications you have to be very careful so that you do not oversimplify and remove security-critical options and hide them where no one sees them. In my low-fidelity prototype, it was out of my scope to decide what options can be removed and what should stay. This would have required extensive domain knowledge. Instead, I settled for presenting another design to group and present the options and let the developers and customer figure out what can be hidden and what should always be visible. The design involved using a list instead of tabs. The benefits of a list system are that you can use scroll more naturally and have more entries. I also suggested a "Simple View" and an "Advanced View" to hide options that are less important. Both versions can be seen in figure 4.7 and 4.8.

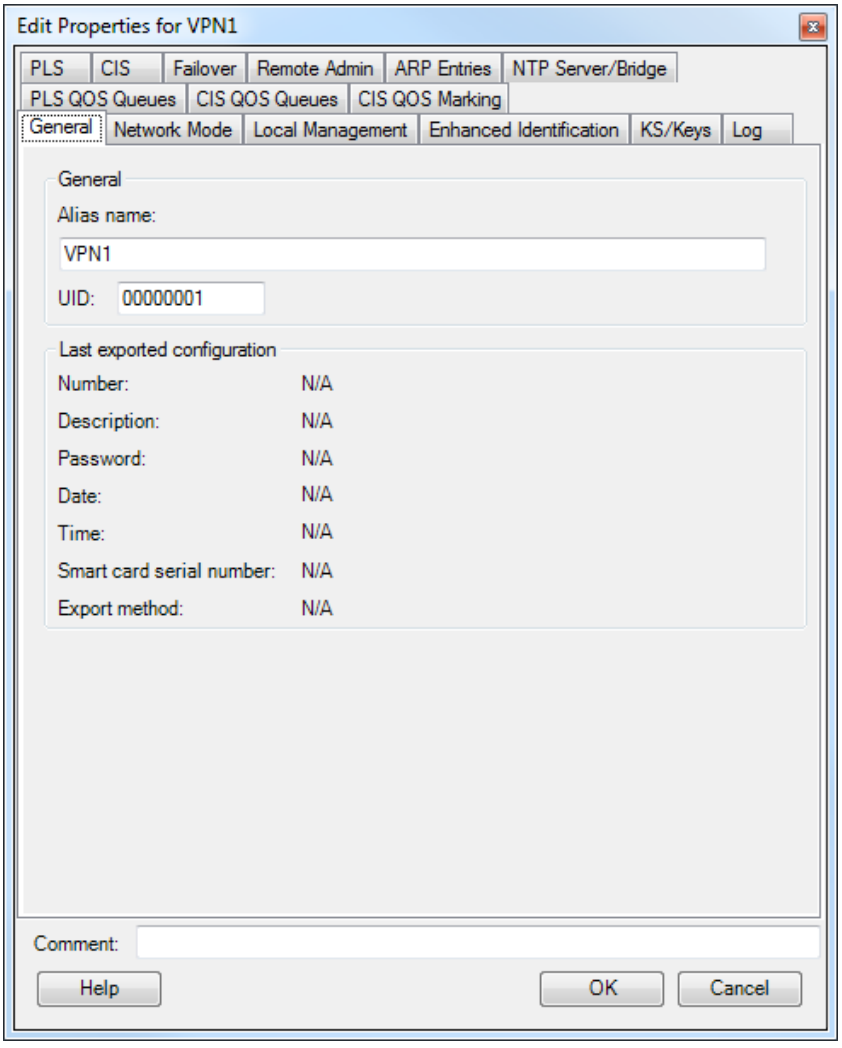

Figure 4.6: Original Properties Design

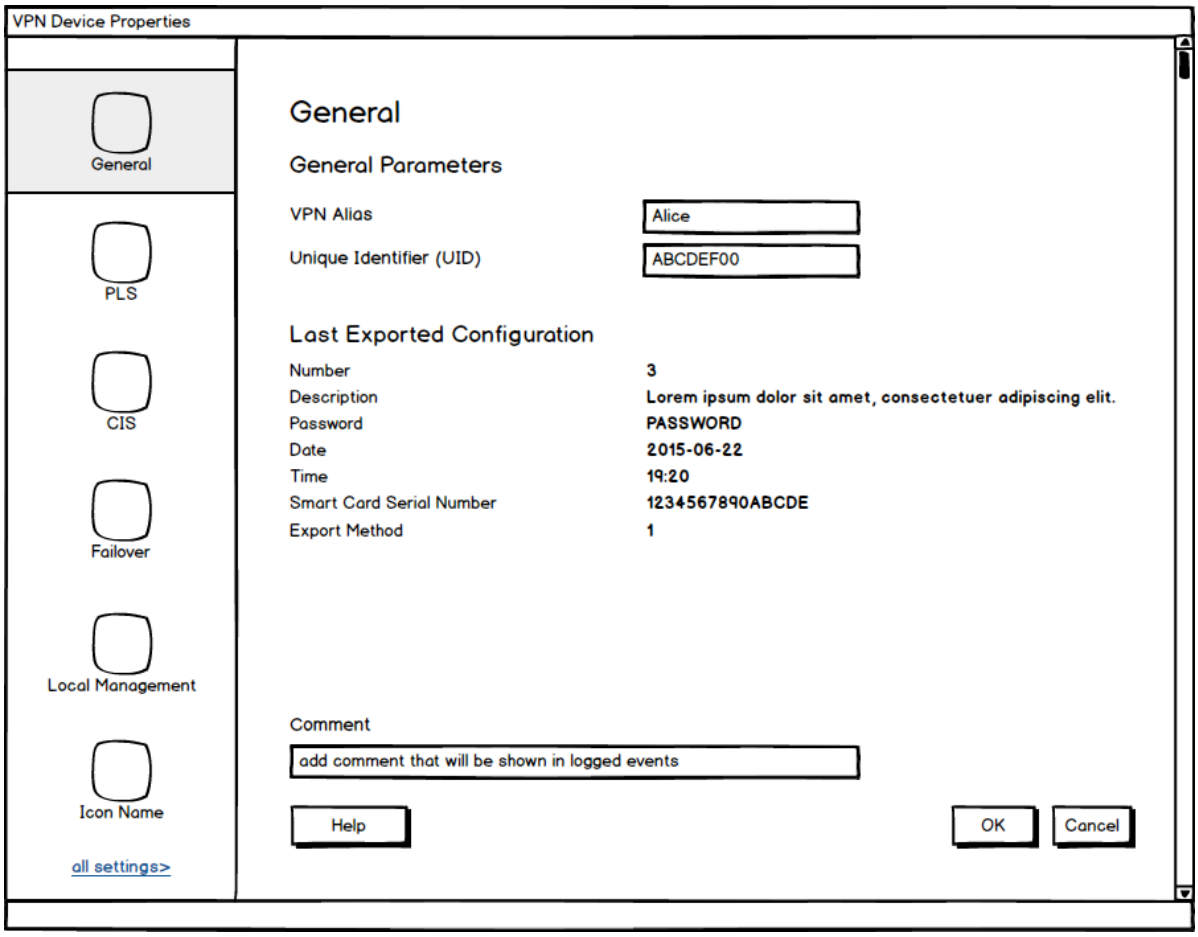

Figure 4.7: Low-Fidelity Properties Design - Simple

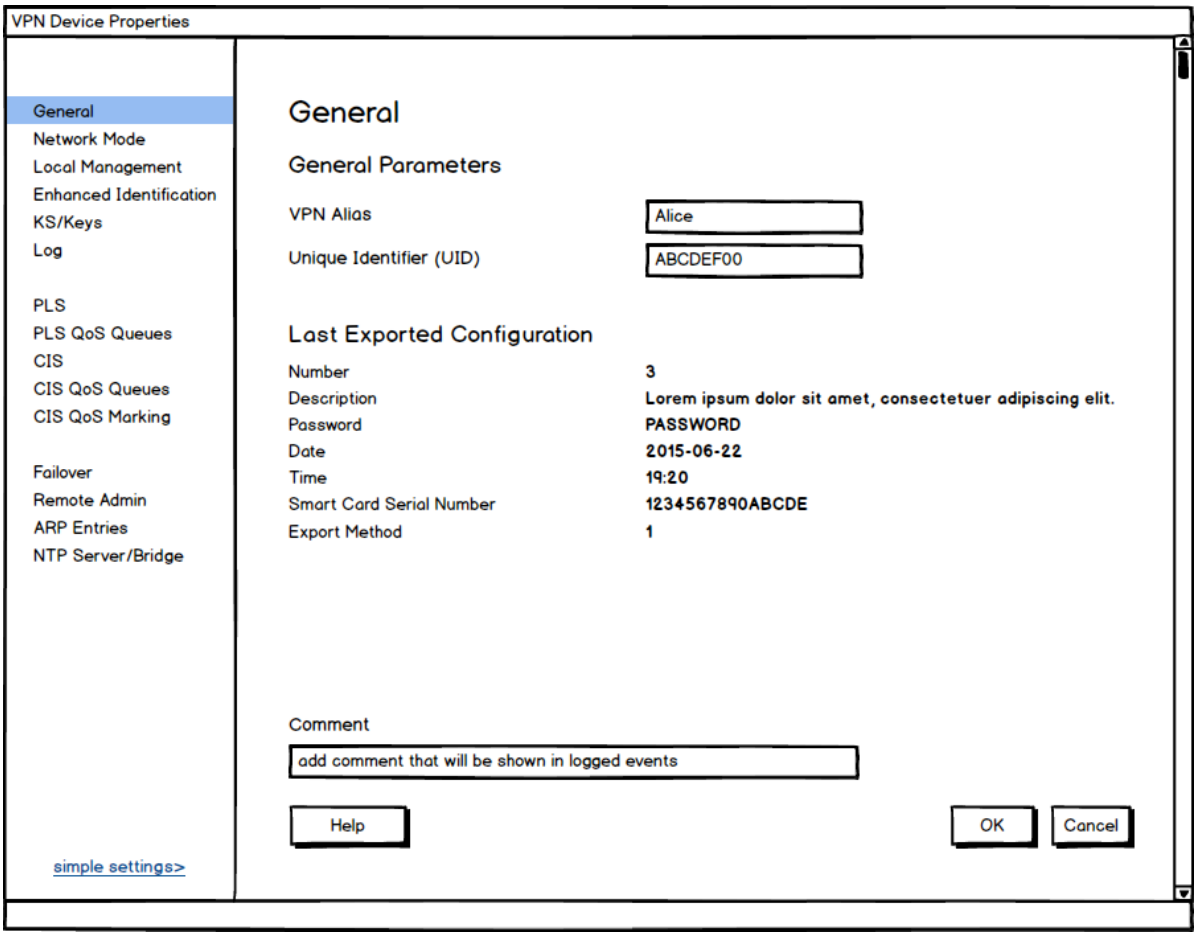

Figure 4.8: Low-Fidelity Properties Design - Advanced

4.3 Design Problem Three: Failover Window

To understand the problem with the failover window, we must first understand what the failover window does. Failover is a mechanism that automatically switches to a backup device that is on standby when the main device fails.

In our case, if the main VPN device fails we want it to switch over to a backup VPN. All VPN devices have a pair of PLS and CIS physical IPv4 addresses configured to them. This is done directly on the VPN device physically. When configuring a failover in the ConfApp configuration application you first have to set up the main VPN device, enter its PLS and CIS IPv4 addresses. Once you have set that up you can go to its options and enable failover. When enabling failover you have to enter new PLS and CIS IPv4 addresses for the main device and you have to enter the PLS and CIS IPv4 addresses for the backup device. The IPv4 addresses you entered for the main device before you enabled failover becomes a virtual address and is used to make the main and backup look like one unit.

This concept is already difficult in itself to understand and it is even a greater challenge to convey this to the user in a user interface. None of the users managed to complete this task. Initially, they did not understand the concept and could not find the option for setting up a backup. After they were shown where the option for failover was they still did not manage to configure it correctly.

Looking at the failover window there are a few problems I believe made this difficult for the user. First off, there is no mention of *virtual* IPv4 address and no indication that you have to reconfigure new PLS and CIS IPv4 addresses for the main VPN. Secondly, the table view does not convey to the user that it needs two pairs of PLS and CIS addresses both for the main device and a second pair for the backup device. The table view seems to accept an arbitrary amount of entries. The test users that tried to configure a failover only entered one pair of PLS and CIS addresses for the backup device. They did not reconfigure for the main device.

In my new design, I tried making the option easier to find. I also thought that the new properties window helped with that, where I went from a multi-level tab system to a list. I also believed it would help if I placed the option to set up a failover in the context menu for the VPN device in the tree view in the main window. You could already set up a parallel VPN device in the context menu and it only seemed natural that you should also be able to set up a failover mechanism in the context menu.

4.4 Design Problem Four: Labels

General improvements were made to the headings and labels. I tried to keep it consistent and separate headings better by using a larger font and use bold text (see figure 4.9). I, again, used the principle of similarity and contrast. Furthermore, this tied back to the complaint about information overload from the test users. By making all text more scannable you reduce the cognitive overload [3]

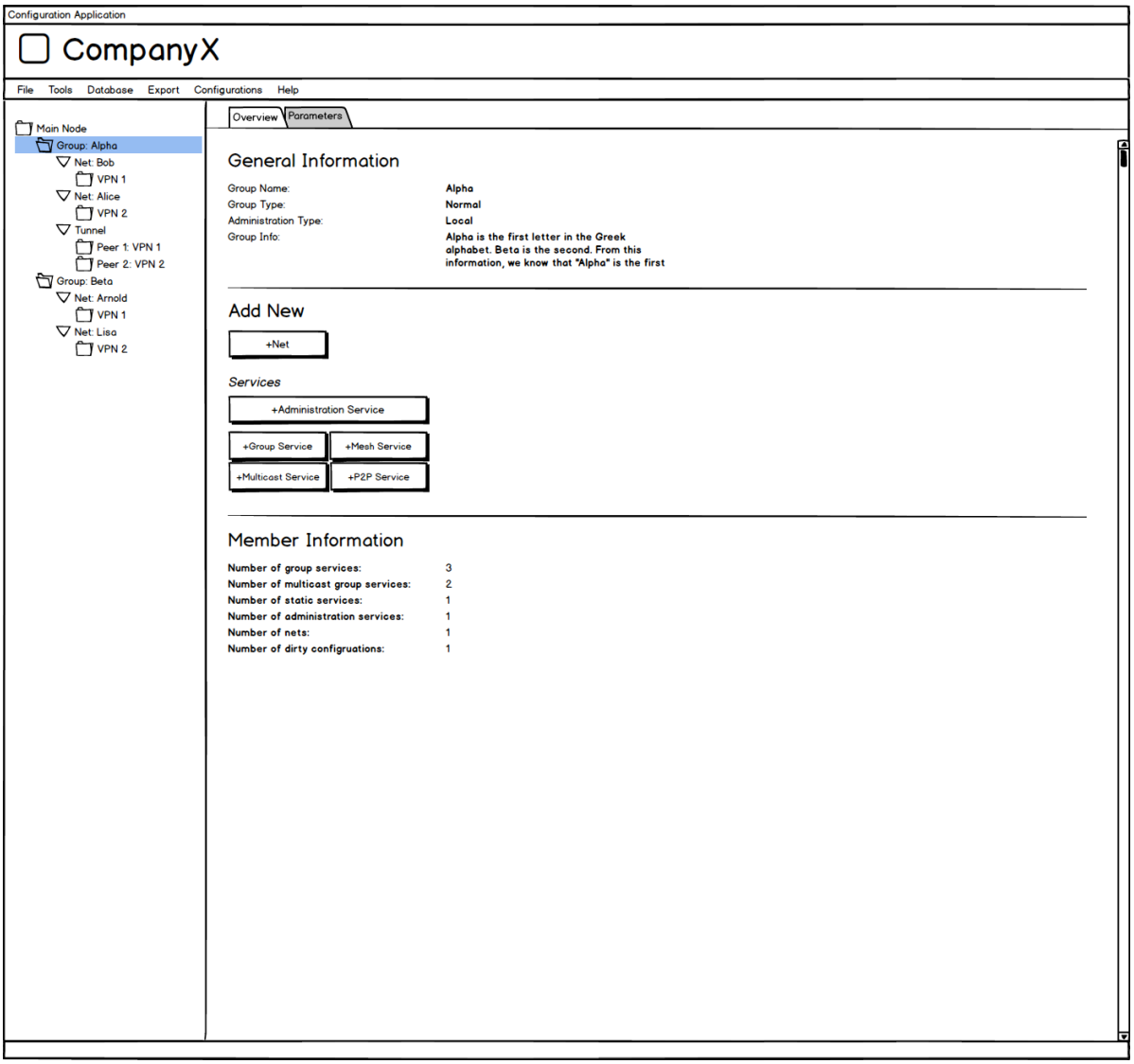

Figure 4.9: Low-Fidelity Group Overview Design

4.5 Design Problem Five: Tunnel Overview

The main purpose of the original configuration application is to allow different networks communicate with each other in a secure way. To accomplish this, you set up tunnels in the application between different networks/VPN devices. The tunnel overview in the original configuration application consists of a table with more than 15 columns of different parameters for each tunnel.

Users struggled with this overview because it lacked affordance. Affordance refers to the perceived and actual properties of a thing, primarily those fundamental properties that determine just how the thing could possibly be used [1, p. 9]. The rows in the table view did not afford editing, see figure 4.10. The table view is in the bottom right and extends for even more columns that have been cropped in the figure to get a better zoom. The users also found that seeing that many variables for each tunnel jarring.

I made the design decision to hide many parameters and if users wanted to see these they had the option of pressing "Details" (see 4.11). The option of editing a tunnel was made more visible by adding an "Edit"-button. I also decided to categorize tunnels in "Enabled" and "Disabled". A search bar was added to make it easier to find tunnels if you have many tunnels set up.

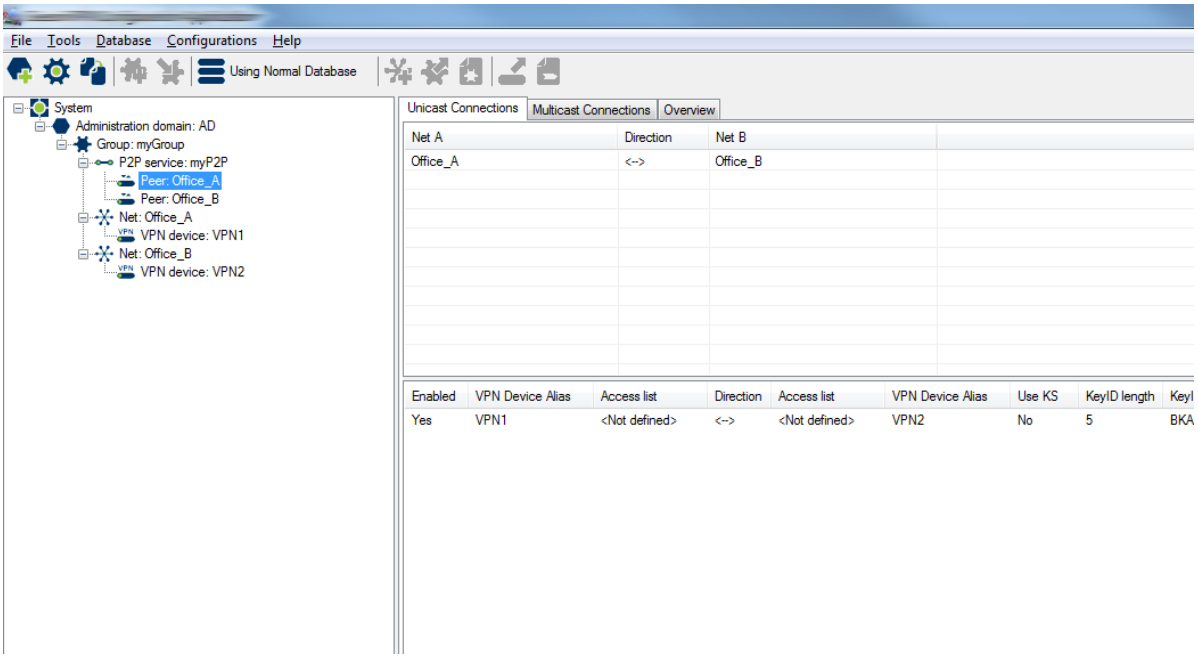

Figure 4.10: Cropped Original Tunnel Overview Design

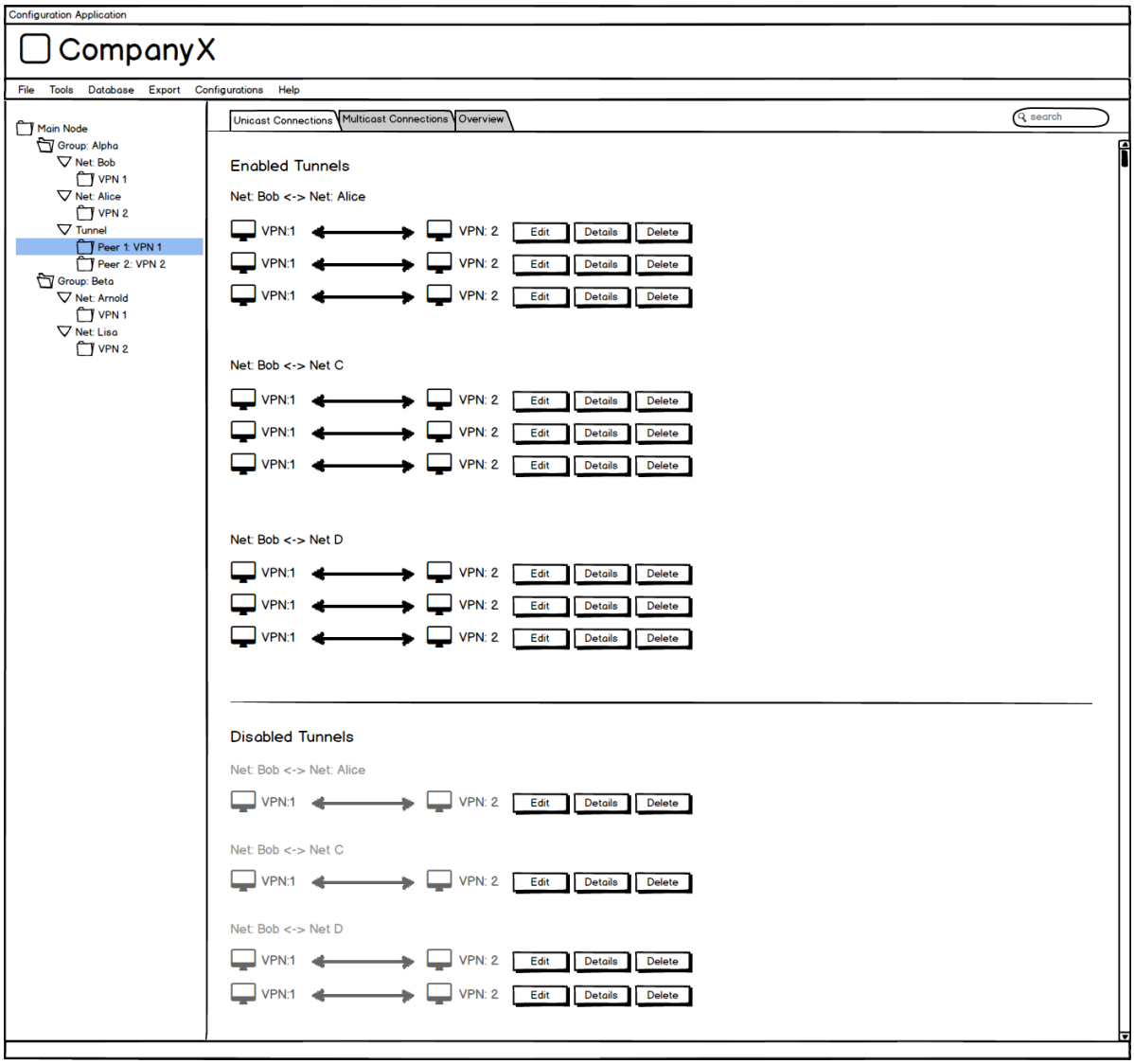

Figure 4.11: Low-Fidelity Tunnel Overview Design

4.6 Design Problem Six: Making Operations More Visible

Context menus are great for providing item specific operations but should not be the only available way to execute these operations. Many of the context menu operations in the original application are only available in this form. Providing an alternative way that was more obvious to the user was one of the goals.

I decided for a simple solution by adding buttons in the right-side pane (see figure 4.9). Not all operations were made available though because that would make the right-side pane extremely busy. I would have liked to make all operations in the context menus more visible, but I estimated it would have required a big design overhaul to accomplish this.

CHAPTER 5

HIGH-FIDELITY PROTOTYPING

I wanted to create a high-fidelity prototype so I could do another usability test. Low-fidelity prototypes are not that particularly well suited for usability tests as they lack affordance, feedback, and you can not perform operations on them.

5.1 Tools

I examined various tools to create high-fidelity prototypes. The current ConfApp application is mostly written in Java using the old Swing package for the user interface.

5.1.1 HTML5, CSS3, and JavaScript

The advantage of using HTML5, CSS3, and JavaScript for creating prototypes is that they are modern, flexible and have extensive support and features. With enough knowledge, most applications and prototypes can be created using these tools.

HTML5 and CSS3 have a wide array of front-end frameworks with Twitter's Bootstrap being the most popular [20]. CompanyX has another product that is using Java as a back end with Twitter's Bootstrap as front end. I considered using the same tools to create a front end to get a consistency for the company. The challenge with this was that original application is an older application with a layout and work pattern that does not lend itself too well with a web application. Some notable things I considered to be issues were:

- Extensive use of context menus
- Multiple windows (e.g. Properties and Wizard windows)

I believe using front end frameworks like Bootstrap is superior with a modern design. If the original application was to undergo a major overhaul with a new requirement specification document then Bootstrap would definitely be an option to be considered.

5.1.2 JavaFX

"JavaFX is a set of graphics and media packages that enables developers to design, create, test, debug, and deploy rich client applications that operate consistently across diverse platforms" [21].

JavaFX is the intended replacement for Swing in Java. It is used to create desktop applications and Rich Internet Applications [21]. I saw this as a natural next step for CompanyX's application. JavaFX provides various components to create UIs with, such as buttons, lists, tree views, text fields and more (see 5.1). JavaFX also provides support for styling components with CSS. I believed JavaFX was the best tools for me to create a mockup with because I already had extensive knowledge of Java and it enabled me to create fully interactive mockups.

5.1.3 Alternative Prototype Tools

The downside of tools like JavaFX and HMTL5, CSS3 and JavaScript is that they require more time to implement. You have to code your user interface yourself. Alternative tools like ProtoIO and Axure are generally faster as they do not require any coding and are only drag and drop style of prototyping. I wanted a more extensive and realistic example of how the application could look, therefore I decided to not use these tools.

5.2 Usability Test of High-Fidelity Prototype

In the usability test of the high-fidelity prototype I aimed to make it as similar as possible to the initial usability test I did on the original configuration application. This to see if users struggled with the same things as they did before.

5.2.1 Challenges of Usability Testing Mockups

5.2.2 Test Participants

I used four out of the five test participants that were used in the original usability test of the original application. In an ideal world, I would have liked to have five new test participants but due to the limitations of the company I used the four old test participants. The last one did not have time to test the prototype.

Due to this limitation I could not directly compare the two usability tests. Instead, I tried to let the users use the application, see what errors they made with the new prototype, and asked them for their opinions at the end of the test. I wanted to see if the users made the same errors as in the first test.

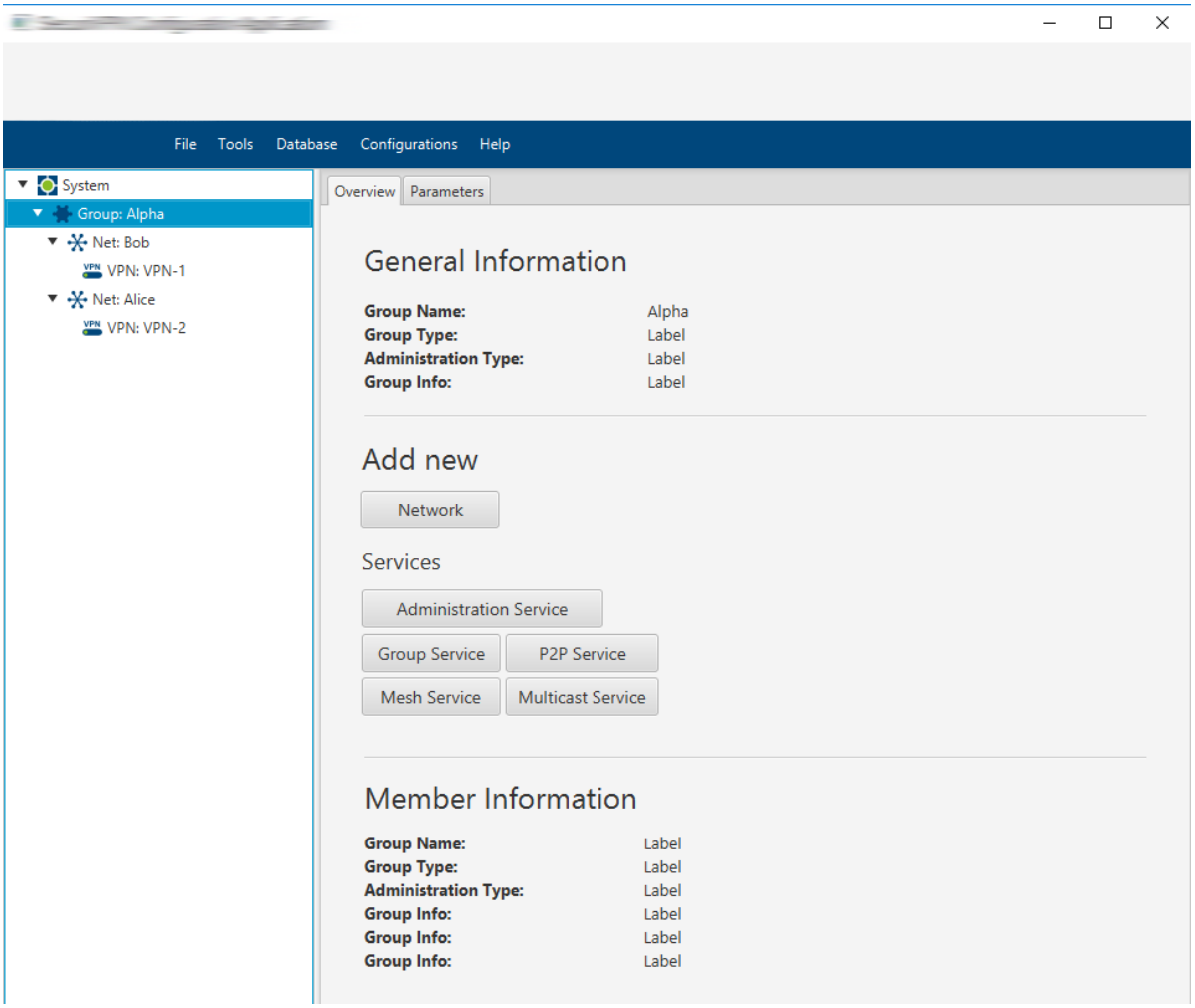

Figure 5.1: JavaFX Group Overview (company logo removed)

5.2.3 Scenario and Tasks

For this usability test I used the same scenario as the initial usability test and I used tasks 1-4 (see table 3.1). I decided not to use task 5 and 6. The reason for this was that users did not struggle with those tasks and they did not make any security-critical errors. The tasks were also relatively simple and the operations would have looked nearly the same in my high-fidelity prototype. See Appendix B for the data that was used.

5.2.4 Results

In this section, I try to summarize the most significant results from the usability test from the goals I laid out at the beginning of the chapter.

5.2.4.1 Successful Task Completion

Figure 5.2 shows the number of participants that successfully completed tasks 1-4.

Figure 5.2: Amount of people who completed each task.

5.2.4.2 Security-Critical Errors

Two out of four test participants failed to set up a backup VPN (i.e. failover). They failed to understand the concept of having to reconfigure the main VPN PLS and CIS IPv4 addresses. Three out of four test participants managed to find the failover option and the one that failed managed to find it once he realized that the context menus still existed.

5.2.4.3 Error-Free Rate

None of the participants were able to complete any task with zero critical or non-critical errors.

5.2.4.4 Likes, Dislikes, and Recommendations

The users expressed a general liking for the new prototype. They felt it was a natural step for the ConfApp application.

The test users all noted that they felt like one of the problems with this test was that the prototype felt more streamlined and lacked functionality and information that the original configuration application had. They said that the context menus felt more clean and lacked operations compared to the original. What they did not realize was that all menu options that were in the original application were also there in the prototype, just not all of them were implemented. So I could still see if they clicked on those options. When I showed the users that all options were still there they were surprised. They said again that the prototype felt much more streamlined and that they had an easier time knowing where to go next. This might be attributed to the design choice of soft staging and keeping it consistent.

When asked what they thought could be improved they generally expressed that they were happy with how the prototype worked. The major complaints were with the failover window. They still felt it was confusing. They said it was difficult to understand that they needed to reconfigure the main VPN device. Two users felt that all context menu options should have been accessible somewhere else. I agree with this sentiment and I mentioned in in the low-fidelity prototype chapter. One user said the button rows that were added could have been better grouped and also suggested graying out buttons that could not be used due to a precondition that had not been fulfilled yet, which I agree with.

CHAPTER 6

DISCUSSION AND CONCLUSIONS

In this section I will discuss my test results. I will also discuss my low-fidelity and high-fidelity prototypes and come back to the points about security and usability, how they work together, and my conclusions.

6.1 Usability and Security

The original application suffers from some of the problems many security-critical applications do; they are complex and difficult for people to use. I have shown that many of security and usability issues in the original application can be solved with existing methods.

But my major concern after working on this project is: How do we make usable software without having any contact with the end-user? Security companies that deal with the highest level of security levels have limited contact with their end-user. This master's thesis was also limited in that way because they also do not want to share all their trade secrets and make them public.

6.2 Domain Knowledge and Usability Test Issue

Creating usable software in the security domain requires domain knowledge. It is crucial to have this to be able to understand the security-critical errors that can happen. It is also important to understand the end-user, his goals and how he works. I did not have contact with the end-user, but I also realize that with these kinds of applications the end-user is often trained by the company of the product on how to use the application. I needed to understand the intended workflow and how I wanted people to use the application and that information could somewhat be acquired by talking to the company.

At the end of this master's thesis work I was informed by one of the company's security experts that I had misunderstood what hash length in the ConfApp application meant. Hash length is *not* encryption. This does invalidate the problem where the user error in our tests lead to a security-critical error but does not take away the fact that our users still had trouble finding the hash length. They also did not understand what hash length meant and I did not either. This could be seen as an issue because I did get a walkthrough by the company and still confused *hash length* and *encryption*.

6.3 Test Participants Challenges

As mentioned in the previous section, domain knowledge is definitely something I value. Finding people that have domain knowledge in how networks and data communication work is not something that is easy to find. It is a specialized area. I was limited in which test persons I could get and while it was not perfect I still think I found some of the major usability problems with the tasks that were set up.

Prerequisite training is necessary to achieve this because security applications generally use very domain specific terms and options so the average computer person without prerequisite training would have no idea how to solve the tasks in my test.

6.4 Low-Fidelity Prototyping Challenges

There is an argument for using paper prototypes and similar techniques to quickly get mockups and ideas of how to design our redesign. I, however, felt that paper prototypes do not suit all people and felt that paper prototypes make it hard to imagine how the end result of the new redesign will look. Balsamiq Mockups felt more realistic and gave a better picture of how the end result would look like.

In an ideal world, I would have liked to test the low-fidelity prototypes too but as always in projects in the real world you are limited by resources. Testing the low-fidelity prototypes would have given us the opportunity to notice errors earlier and fix small design mistakes.

At the start of the low-fidelity prototype stage, I planned on making a major overhaul of the ConfApp application. I planned on looking for alternative ways the application as a whole could be designed and used. I wanted to make it more streamlined. However, after receiving the requirement specification the company used that was created by their customer I realized I would be working under many constraints.

The customer organization were extremely specific in their requirement specification. They wanted groups and they wanted them to be called groups. They had requirements on how they wanted things to be set up, for example, they wanted to use services (P2P, Mesh, Group, and Multicast) as a tool to create tunnels. Due to the many requirements I felt limited in how big of a redesign I could do and instead decided that I would try to keep the same overall structure and workflow of the ConfApp application and instead

focus on minor improvements and try to find solutions for the security-critical errors I encountered in my initial usability test.

6.5 High-Fidelity Prototyping Challenges

Creating high-fidelity prototypes when you already have elaborate low-fidelity ones is easy. It is just time-consuming to implement everything with a tool like JavaFX. I was overall happy with the end result. I felt it was a realistic prototype that I could see being used in the future. I tried creating the same look and feel as the company's other products. Additional styling can be done through CSS.

6.6 Validity of User Testing

To avoid user errors when using security-critical applications we need to use usability evaluation methods. User testing is one of the most popular ones. Looking back, we have to ask ourselves if this method is suitable for finding usability errors in security-critical applications where the end-user is not available as a tester. Do the constraints when working at a security company hinder us from using usability testing as an evaluation method?

The challenge with not having the end-user as a tester is understanding who the end-user is. Can we find people that can represent these end-users? There are well-established methods to deal with this in the form of prerequisite training. But another challenge arose when I wanted to introduce this method; It required domain knowledge in an already complex application. While I had basic knowledge of computer security and adequate knowledge in network communication I still had to spend time learning how the configuration application was used. I feel that it is difficult for the developer to also be a usability expert and a usability expert without the expertise in how the system works like the developer has can create some problems. But even with the problems mentioned I feel I found a lot of usability issues with the tasks I created.

6.7 Usability and Information Security

The original application is a product that enforces information security. In my case, I looked at Data Integrity, Confidentiality, and Availability.

Data Integrity

I looked at access lists as a way to maintain data integrity. If access lists are set up wrong it could cause potential disasters. Unauthorized people might be able to access top secret devices and information. We need to ask ourselves if access lists is a well-understood concept? Is there in any way we can make it clearer how access lists work? Is the way we represent access lists in today's security software as good as it can be?

Confidentiality

How do we protect our system and devices? A standard way to do this is to use encryption, hash length and passwords/keys. Our users failed to understand what hash length meant. As I also mentioned, I confused encryption with hash length. Is that something that should be attributed to lack of knowledge or an issue with the application?

The application also used shared keys and key masks. In our test most users understood what a key was but there seemed to be a confusion in what a key mask was and how to set it. Is this just a concept that has to be learned or is there a way to make it clearer what a key mask is? Another issue with this is that the key mask is set by the user. Is this the best way to handle keys? Putting the burden on the user and having him remember how keys work and manage his own keys.

Availability

With the customers CompanyX have we can expect that they have high requirements on availability. In my usability test, the most prevalent problem was that all users failed to set up a backup VPN/VPN failover mechanism. Failure to do so can lead to unfortunate consequences. What do we do when a device that we rely upon does not work and we have no backup?

I have discussed this a bit earlier, but I feel this part of the application could use some major improvements. The concept of parallel VPN device and a failover mechanism seemed to cause confusion for our users. But even after they were told it was a failover they wanted they did not manage to set one up. Fortunately, the application does manage to tell the user if they have not set it up properly. Error prevention is an important feature for all types of applications, but especially for those that deal with security. If a user fails to do a task in the correct way they need to be notified in some way that makes them realize their error.

6.8 Future Work

This thesis presented usability problems when dealing with a security-critical application. Additional work with the application can be done. I did not look at all the features of the ConfApp application. You can further improve on the security mechanisms in the application and try to find more security-critical errors. Additional user testing or heuristic evaluation can be done to find more usability problems.

6.8.1 Design Patterns

I did not create any general design patterns that can be used when designing security-critical applications. There are plenty of guidelines and design patterns for everyday applications and websites but not many in the security domain. A few examples could be looking at how to design access lists, how to represent encryption and how to handle key management.

6.8.2 Design Processes

At the start of this thesis work I wanted to look at design processes for creating usable security-critical applications but with how the thesis work progressed I decided to focus on usability problems in an existing product. The AEGIS process described in the theoretical background section shows an example of a design process pattern. I could have looked at how to assess security risks and how they tie to usability and the users. How do you communicate with the customer at the start of the application development process?

6.8.3 Finalize Prototype

Create a fully working prototype that can be deployed to the end-user and get real data on how it is used. Potential major overhaul of the prototype could also be done with a more staged workflow and more visible controls instead of using context menu controls.

6.9 Conclusions

Usability in security-critical application development presents problems that there have not been much research about how to solve. How do we represent access lists? What terminology should be used for consistency? How do we deal with domain specific mechanisms like failover and tunnel negotiation? A lot of knowledge is required to find usability errors that affect security. It requires your usability tester to have both knowledge in the user-centered design field and the information security field. Furthermore, many of the usability issues that exist in everyday software still exist in security-critical software. In this thesis, I tried to find ways to elicit usability issues with the constraints that were set up by having to deal with security-critical software development. There is a need to prioritize usability. I believe by raising the awareness of the potential dangers that can arise with user errors, I can make more people in the information security domain to start realizing how important usability is. I have shown in this thesis some of the security-critical errors that can occur when usability is not considered enough. I also provided simple example solutions to the problems I found.

When designing applications that are supposed to help users uphold information security or any type of security we need to *enable* the user. To accomplish this, we need to make security usable. It might not always be possible to make *all* security usable but if we do not even consider usability as a factor we are by default disabling the user.

BIBLIOGRAPHY

[1] D. A. Norman, *The Design of Everyday Things*, reprint ed. Basic Books, Sep. 2002.

[2] J. Nielsen. (1995, Jan.) 10 usability heuristics for user interface design. [Online]. Available: http://www.nngroup.com/articles/ten-usability-heuristics/

[3] ——. (1995, Jan.) How to conduct a heuristic evaluation. [Online]. Available: http://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/

[4] S. Bradley. (2014, Sep.) Design principles: Connecting and separating elements through contrast and similarity. [Online]. Available: http://www.smashingmagazine.com/2014/09/design-principles-connecting-and-separating-elements-through-contrast-and-similarity/

[5] S. Sen and S. Samanta, "Information security," Retrieved 2015-09-15. [Online]. Available: http://www.ijirt.org/vol1/paperpublished/IJIRT101673_PAPER.pdf

[6] I. Flechais, C. Mascolo, and M. A. Sasse. Integrating security and usability into the requirements and design process. [Online]. Available: http://www.cl.cam.ac.uk/~cm542/papers/icges.pdf

[7] I. Fléchais, "Designing secure and usable systems," Ph.D. dissertation, University of London, Feb. 2005. [Online]. Available: http://www.cs.ox.ac.uk/files/6345/thesis_final.pdf

[8] J. Nielsen. (2012, Jan.) Usability 101: Introduction to usability. NN/g.

[9] S. Krug, *Rocket Surgery Made Easy: The Do-It-Yourself Guide to Finding and Fixing Usability Problems*, 1st ed. New Riders Press, Dec. 2009.

[10] J. Rubin and D. Chisnell, *Handbook of Usability Testing: Howto Plan, Design, and Conduct Effective Tests*, 2nd ed. Wiley, May 2008.

[11] P. K. Chilana, J. O. Wobbrock, and A. J. Ko, "Understanding usability practices in complex domains," pp. 1–2, 2010. [Online]. Available: http://dub.washington.edu/djangosite/media/papers/pap0992-chilana.pdf

[12] J. Nielsen. (2000, Mar.) Why you only need to test with 5 users. [Online]. Available: http://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/

[13] Open broadcaster software. Retrieved 2015-09-15. [Online]. Available: https://obsproject.com/

[14] Planning a usability test. Retrieved 2015-09-15. [Online]. Available: http://www.usability.gov/how-to-and-tools/methods/planning-usability-testing.html

[15] Red/black concept. Retrieved 2015-10-20. [Online]. Available: https://en.wikipedia.org/wiki/Red/black_concept

[16] Balsamiq mockups. Retrieved 2015-09-15. [Online]. Available: https://balsamiq.com/products/mockups/

[17] A. Whitten, "Making security usable," Ph.D. dissertation, Carnegie Mellon University, May 2004. [Online]. Available: http://www.gaudior.net/alma/MakingSecurityUsable.pdf

[18] J. Nielsen. (2007, Sep.) Tabs, used right. [Online]. Available: http://www.nngroup.com/articles/tabs-used-right/

[19] ——. (2011, Jan.) Top 10 mistakes in web design. [Online]. Available: http://www.nngroup.com/articles/top-10-mistakes-web-design/

[20] Twitter's bootstrap. Retrieved 2015-09-15. [Online]. Available: http://getbootstrap.com/

[21] JavaFX: Getting started with JavaFX. Retrieved 2015-09-15. [Online]. Available: http://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-overview.htm

Appendix A

Data

Office A's Network

IPv4 address: 212.116.87.0
Netmask: 255.255.255.0 or /24

VPN

Plain Text Interface (PLS): 212.116.87.2/24
PLS Gateway: 212.116.87.1
Cipher Text Interface (CIS): 214.185.0.10/24
CIS Gateway: 214.185.0.1

VPN Backup 1

PLS: 212.116.87.3
CIS: 214.185.0.11

VPN Backup 2

PLS: 212.116.87.4
CIS: 214.185.0.12

Computers

Computer 1: 212.116.87.5
Computer 2: 212.116.87.6
Computer 3: 212.116.87.7
Computer 4: 212.116.87.8

Office B's Network

IPv4 address: 212.116.80.0
Netmask: 255.255.255.0 or /24

VPN

PLS: 212.116.80.2/24
PLS Gateway: 212.116.80.1
CIS interface: 214.175.0.10/24
CIS Gateway: 214.175.0.1

VPN Backup 1

PLS: 212.116.80.3
CIS: 214.175.0.11

VPN Backup 2

PLS: 212.116.80.4
CIS: 214.175.0.12

Computers

Computer 1: 212.116.80.5
Computer 2: 212.116.80.6
Computer 3: 212.116.80.7
Computer 4: 212.116.80.8

Appendix B

Data

Office A's Network

IPv4 address: 1.1.1.1
Netmask: 255.255.255.0 or /24

VPN

Plain Text Interface (PLS): 1.1.1.2/24
PLS Gateway: 1.1.1.3
Cipher Text Interface (CIS): 1.1.1.4/24
CIS Gateway: 1.1.1.5

VPN Backup 1

PLS: 1.1.1.6
CIS: 1.1.1.7

VPN Backup 2

PLS: 1.1.1.8
CIS: 1.1.1.9

Computers

Computer 1: 1.1.1.10
Computer 2: 1.1.1.11
Computer 3: 1.1.1.12
Computer 4: 1.1.1.13

Office B's Network

IPv4 address: 2.2.2.2
Netmask: 255.255.255.0 or /24

VPN

PLS: 2.2.2.3/24
PLS Gateway: 2.2.2.4
CIS interface: 2.2.2.5/24
CIS Gateway: 2.2.2.6

VPN Backup 1

PLS: 2.2.2.7
CIS: 2.2.2.8

VPN Backup 2

PLS: 2.2.2.9
CIS: 2.2.2.10

Computers

Computer 1: 2.2.2.11
Computer 2: 2.2.2.12
Computer 3: 2.2.2.13
Computer 4: 2.2.2.14

ConfApp Screenshots

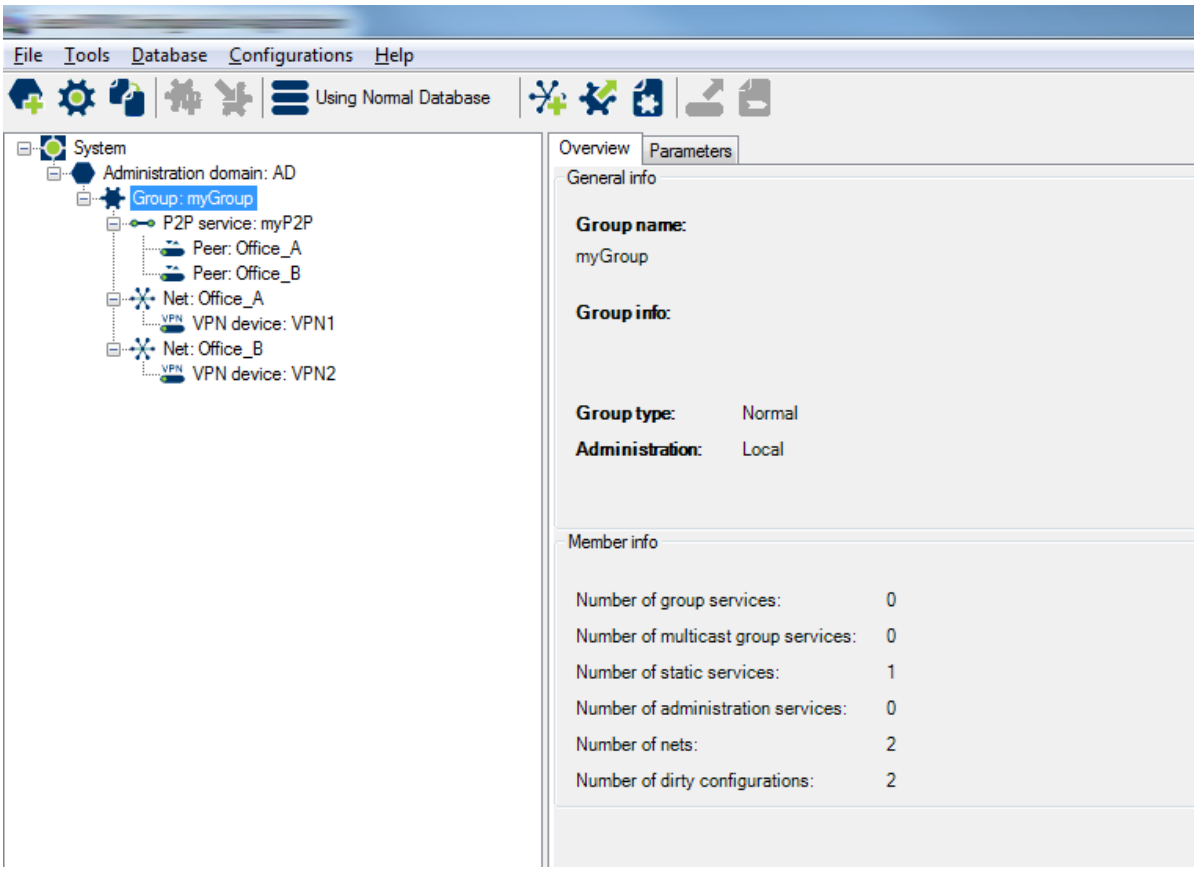

Figure 1: Cropped Group Overview

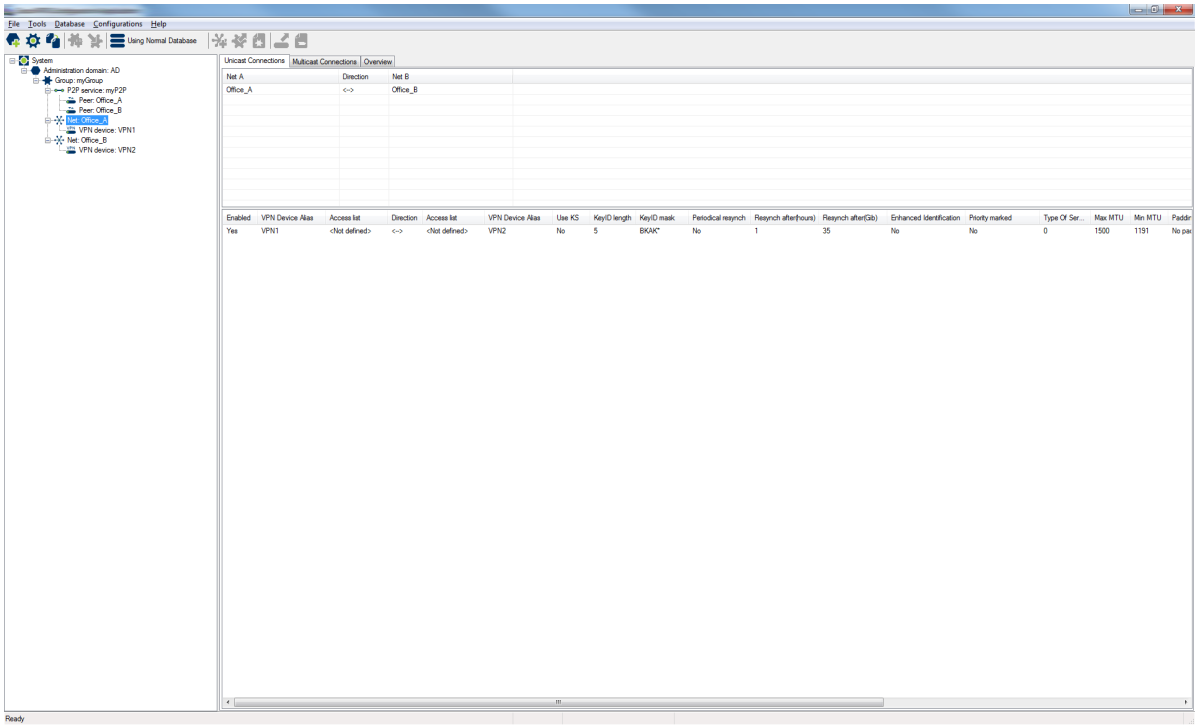

Figure 2: Net Overview

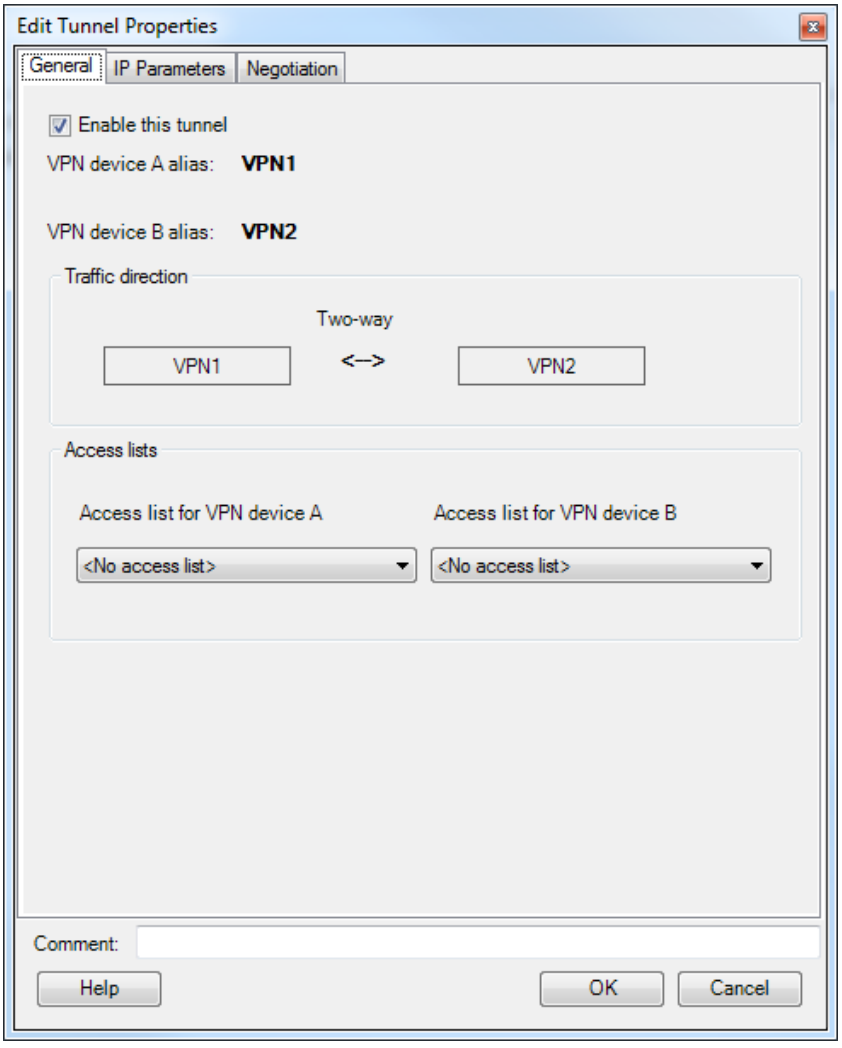

Figure 3: Tunnel Properties

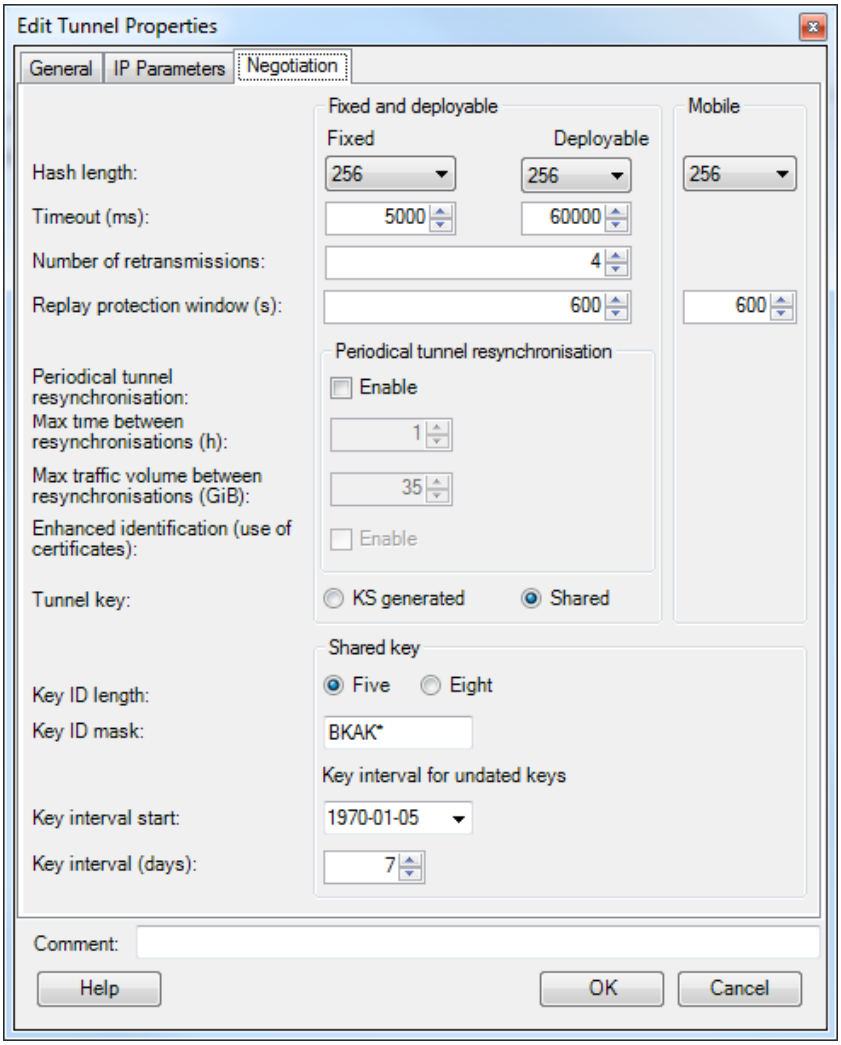

Figure 4: Tunnel Properties - Negotiation

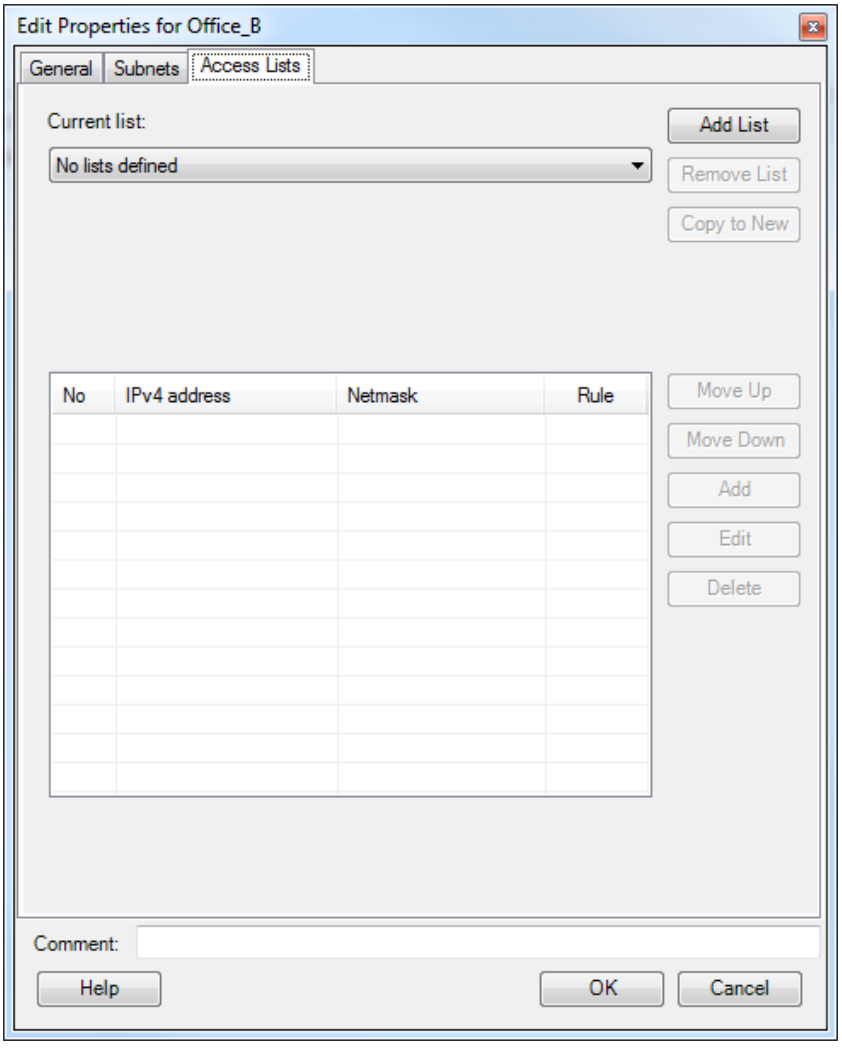

Figure 5: Net Properties - Access List

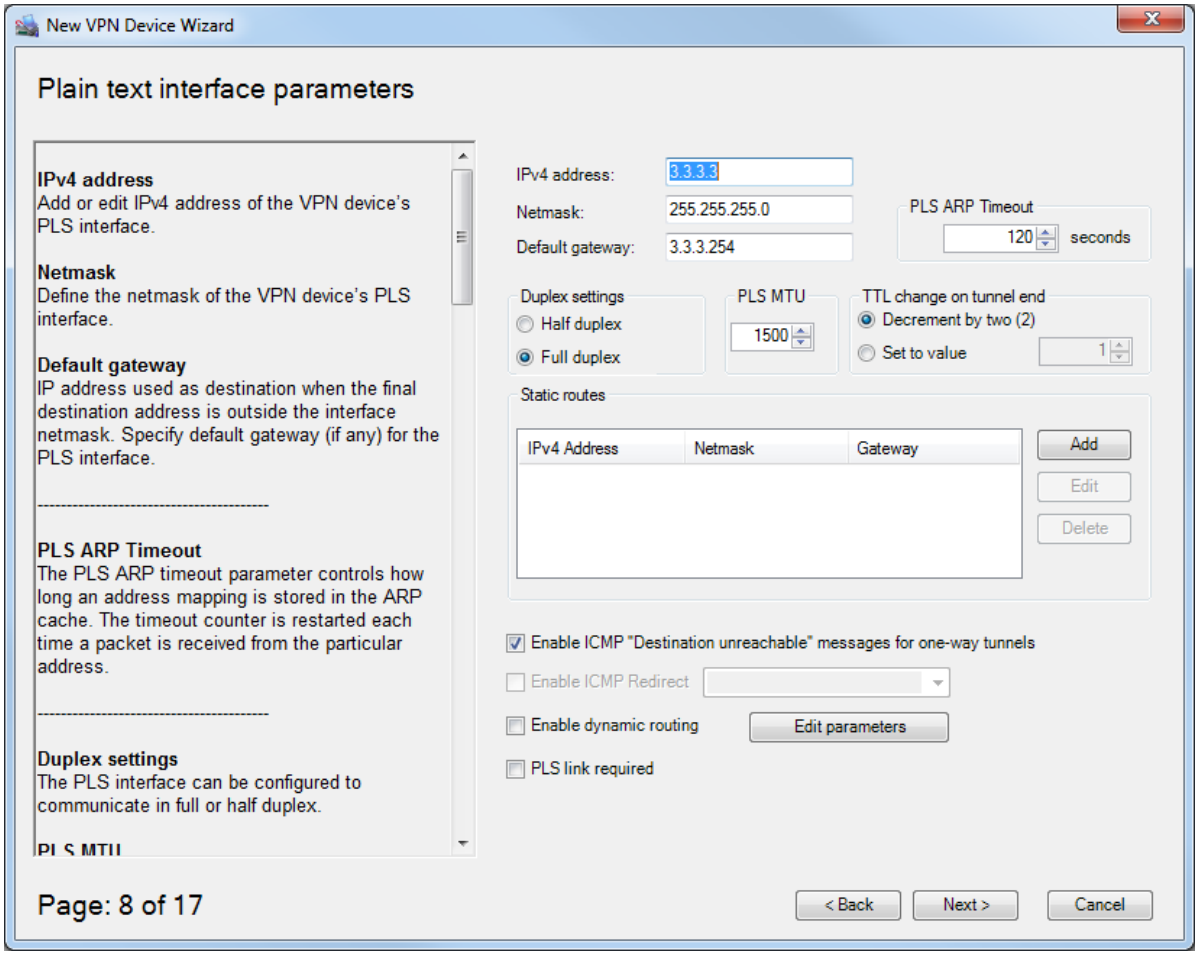

Figure 6: VPN Wizard - Step 3

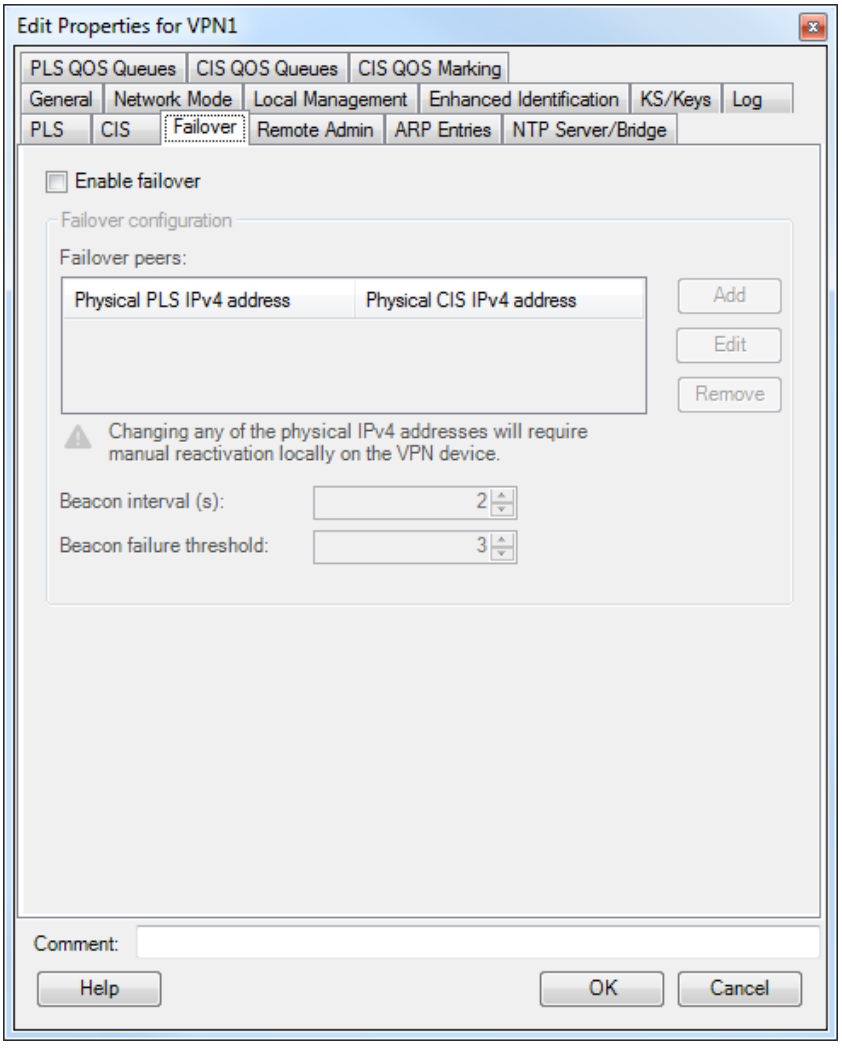

Figure 7: VPN Properties - Failover

APPENDIX D

Screener E-mail

Hello,

My name is Truong Hoang and I am currently writing my master's thesis here at CompanyX about design processes for and usability of user interfaces in security-critical systems. As a part of my thesis, I will be conducting a usability study on the ConfApp configuration application. I am currently looking for people to take part in this usability study.

Prerequisites
The prerequisite for participating is to have a basic knowledge of the ConfApp application. People who have been involved with the development of the configuration application and are experts cannot participate in this usability study.

What will you be doing in a usability study?
You will be asked to do several short tasks using the configuration application and be asked to share your experience and perceptions of the application.

How long is a session? 30-60 min

When and where?
The plan is to do this usability test between May 25th to June 5th during weekdays 08:00-17:00 and it will be done here at CompanyX's offices.

Interested?
Please reply to this email if you are interested with your name and when you are available for the study. The dates are not set in stone and are quite flexible and can be adjusted to your schedule. If you have any questions, please contact me at truong_hoang@live.com

Thank you for reading,
Truong Hoang