Aligning Usability and Security: A Usability Study of Polaris

Alexander J DeWitt
School of Information Systems, Computing and Mathematics
Brunel University, Kingston Lane
Uxbridge. West London UB8 3PH. UK
+441895274000
Alex.Dewitt@Brunel.ac.uk

Jasna Kuljis
School of Information Systems, Computing and Mathematics
Brunel University, Kingston Lane
Uxbridge. West London UB8 3PH. UK
+441895274000
Jasna.Kuljis@Brunel.ac.uk

ABSTRACT
Security software is often difficult to use thus leading to poor adoption and degraded security. This paper describes a usability study that was conducted on the software 'Polaris'. This software is an alpha release that uses the Principle of Least Authority (POLA) to deny viruses the authority to edit files. Polaris was designed to align security with usability. The study showed that despite this aim, usability problems remained, especially when the study participants had to make security related decisions. They also showed apathy towards security, and knowingly compromised their security to get work done faster. This study also demonstrates the difficulty in achieving security and usability alignment when the usability is a post hoc consideration added to a developed product, rather than being integrated from the start. The alleviation of usability problems from security software proposed in this paper are threefold: reducing the burden on the user to make security related decisions, counteracting user's apathy by ensuring that the fast way of doing things is the secure way, and integrating security software with the operating system throughout development.

Categories and Subject Descriptors
D.4.6 [**Operating Systems**]: Security and Protection – *Invasive software.*

General Terms
Experimentation, Security, Human Factors.

Keywords
Usable security, HCI-SEC, Polaris

1. INTRODUCTION
In November 2000, Jakob Nielson said in his online alertbox: "Usability advocates favour making it *easy* to use a system … security people favour making it *hard* to access a system" [13]

This quote best illustrates a long-held belief that security and usability do not go hand-in-hand. Many software designers share the notion that improving security necessarily degrades usability, and vice-versa [23]. Users, on the other hand, believe that being difficult to use is a part of being secure [23, 26]. Having higher levels of security in practice often means extra expense in terms of user time and effort to learn and implement these systems, as well as possible confusion over the level of protection required for the task at hand [6].

We wanted to assess the extent of usability problems that users might encounter when using security software. For that purpose we conducted a detailed usability study on a new software, 'Polaris' [17]. The next section provides some background information on issues in usability of software. This is followed with a section that introduces the Polaris software. A section on a usability study conducted on Polaris follows. The subsequent section reviews the results of the study. In the concluding section recommendations on how to alleviate these problems are made followed by the future research directions.

2. USABILITY OF SECURITY SOFTWARE
Usability has until recently played a relatively minor role in the development of many types of software. In 1988 Boehm introduced a software project methodology known as the spiral model [2]. This model advocated an iterative approach in which each stage is repeatedly evaluated and redesigned in order to achieve a better end product. This allowed usability to be integrated into the design of a software product from the very start. Usability engineering is now a widely practiced activity; however its application to security software leaves room for improvement.

The first mention of usability and security having to work hand in hand is generally accepted to be in the 1975 paper "The protection of Information in Computer systems" [16]. The authors proposed eight principles to guide the design of security products, the last of which was 'Psychological acceptability'. It described the interface design to be essential, so that users routinely use the security mechanisms in the correct way.

More recently, research into usable security leads a school of thought that security and usability can, and indeed should be complimentary to one-another. This research area has become known as HCI-SEC (Human Computer Interaction and Security). Usability is crucial because even the most secure system would not be used much if it is too difficult to learn and cumbersome to use so that users would rather choose to bypass it in order to get their work done. HCI-SEC recommends that usability and security play equal roles throughout the design and implementation of

Symposium On Usable Privacy and Security (SOUPS) 2006, July 12-14, 2006, Pittsburgh, PA, USA.

software [1]. This has not been the case in the past as traditional software development teams have had either security experts, or usability experts, but not both. More collaboration between the two groups of experts would have been better.

Whitten and Tygar [21] performed one of the first, and most well known experiments to test the usability of security software. Their 'Why Johnny can't encrypt' study showed that Pretty Good Privacy (PGP) encryption software, which was considered to have a particularly good interface at the time, was not usable enough to allow users to adequately protect their e-mails. The authors suggested that unlike ordinary software, the usability of security software is not entirely based on the interface design. Garfinkel followed up Whitten and Tygar with his 2005 'Johnny 2' study [9]. This study evaluated the usability of Key Continuity Management as a compromise on security, and found that it is a workable model for allowing naïve users to protect their e-mail. Balfanz and Grinter [1] evaluated the usability of deploying a Public Key Infrastructure (PKI) in 2004. They found that even though the technology was mature, and their participants were well educated in computer security, the task was found to be extremely difficult to complete, with the PKI setup requiring 38 steps, each of which forced the user to make a decision; decisions which the users did not know how to make. Also in 2004, Yan et al. [22] reviewed the usability of password security and memorability, and showed that good education can help users to make a good password choice.

HCI-SEC research frequently discussed the notion of software transparency as a usability indicator. Gerd and Markotten [10] recommend that security software be transparent to improve its usability, whilst de Paula et al. [4] take the opposite stance. Dourish et al. [5] pose the question of whether increased transparency of security reduces the users ability to trust the system. Straub et al. [18] propose a compromise whereby there is maximum transparency at first, gradually allowing users to take more and more control as they become accustomed to the software. As we shall see in section 3, an important aim of Polaris was to have very high transparency.

In 2002 Yee [23] introduced new guidelines for aligning security and usability. One of the principles from these guidelines is The Principle of Least Authority (POLA) [24]. It describes the principle of disallowing a program access to all resources except those that it needs to run. This is not the way that Windows and UNIX systems work. For example, a text document opened in Windows will give all the authority of the logged in user to the text editor program, allowing it to traverse and edit the entire file system. This is undesirable if a virus or Trojan takes control of the program and leverages its power to alter or delete files on the system. Instead, Yee recommends that the POLA principle be used, and the user should designate the abilities he wants the program to have. For example, if he wants to edit a file, he uses a file open dialogue to designate edit capabilities to the program. The program should only be able to write to that file, and that file alone.

3. INTRODUCING POLARIS

Polaris is an alpha release software for Windows XP, developed by researchers at HP, which aims to align security and usability. An important point to note is that usability and security are also blended with legacy, in that Polaris is a retrofit onto Windows, designed to allow its millions of users to continue using their established programs. Polaris is not a complete redesign of an operating system, and as such must comply with design features of Windows.

Polaris is based around the POLA principle described in section 2. Polaris prevents any virus or malicious code from, reading, altering, or destroying files on the system, by severely restricting the authority of software so it can only access the files it needs to run. Using Polaris, applications can be 'polarized', creating a 'tamed' version of that application which is immune to viruses; these are known as 'pets'.

The primary goal of Polaris is to make Windows safer from viruses and malicious code, but it was specifically designed to be highly usable as well as secure. By aligning security with usability, the user is less likely to want to circumvent the security system due to frustration, as the easy way should be the secure way. The developers of Polaris had a specific usability goal; that 'the user shouldn't be aware that Polaris was providing protection' [17], in other words, it should be transparent.

By performing a usability study on Polaris, we can measure its success in its goal to be highly usable as well as highly secure. Successes or problems can then be identified and go on to inform future iterations of Polaris and other software in the HCI-SEC field. The ultimate goal is that security and usability will no longer be two distinct fields, but will work in harmony to produce secure software that is highly usable.

4. THE USABILITY STUDY

Our study was the first formal usability study to examine Polaris. The findings will help inform the Beta release of Polaris, as well as to direct research in the HCI-SEC field on improving the synergy of security and usability in other areas such as encryption and authentication technologies.

The methodology used in this study is similar to that employed in the 'Why Johnny Can't Encrypt' study [21], in that it uses a laboratory test which asks users to perform tasks that include the use of security. This study employed a combination of qualitative and quantitative approaches. The Polaris documentation was also included in the evaluation as it is considered a part of the software package. Users were asked to perform some tasks to simulate the configuration of Polaris. After this the tasks represented ordinary computer usage in which the security features were presented as a side-effect of the primary task. This testing scenario would be much the same as in real situations where users would first be required to set up the Polaris software, but thereafter, we assume, would be more concerned with getting their work done than with configuring security.

This study used three pilot tests to refine the testing procedure, followed by ten participants for the main study. Virzi [19] found that 90% of all usability problems were discovered in a study with ten participants, and the usability expert Jakob Nielsen [14] advocates using only five participants in a study. Using more than ten participants would have a very low ratio of problems discovered to resources expended.

The participants who were student volunteers from the department of Information Systems and Computing at Brunel University, all had a good working knowledge of computers, but no specialist knowledge of security issues or terminology. Before the test, Participants' were simply told "Polaris is designed to protect you from viruses by restricting the authority of applications to access

Table 1. Usability test broken down into tasks

Task #	Task Description	Purpose
1	Identify which of three applications have been polarized	Users need to know whether the application they are using has been polarized or not (equivalent to working in a safe environment or a non-safe environment). This task tests how well the software informs the user of their status.
2	Polarize Internet Explorer	Test the process of making an application safe through Polaris
3	Browse an Internet banking website	Observe behavior when using secure and trusted web sites
4	Check e-mails and follow hyperlinks	Observe behavior when using insecure and not trusted web sites and e-mails
5	Manually add some buttons to the toolbar of Microsoft Outlook which will let you safely open and save e-mail attachments using Polaris.	Test the procedure for adding functionality to safely open e-mail attachments, which must be manually configured in the current alpha release of Polaris.
6	Check e-mails and try out attached files	Observe what security precautions are taken when trying attachments of unknown origin.
7	Download an application from the Internet, and try it out on your system.	Polaris includes several ways to safely try out applications, this task tests which one users tend to take (if any) when they want to try out a potentially dangerous application.
8	De-Polarize Microsoft Word and open a document in the normal, unprotected version of Word.	Test if the user can correctly go back to using the normal version of applications after using Polaris.

your files". Participants were not given any further instruction or training, but were able to consult the documentation in electronic format during the test. The participants understood that they were part of a usability study, but that they should use the PC as they would their own.

During the tests, participants were alone in a room with a PC. They were observed through a one-way mirror by the experimenter in an adjoining room. Their keystrokes and screen activity were captured and stored electronically for later analysis. A list of the tasks users performed during the test can be found in table 1.

At the end of the testing, participants completed a questionnaire to gather subjective opinions. The questionnaire measured results on the System Usability Scale (SUS) [3]. SUS has been shown to be a good overall guide to usability, and has been used extensively within its originating company, Digital Equipment Co. Ltd., and in external studies such as [15].The SUS is designed to give a quick impression of the overall usability of a product. It consists of ten questions rated on a Likert scale, and yields a number from 0-100, where 100 represents excellent levels of usability. SUS was chosen for this study because it is very short and quick to complete. It is believed that this would avoid user frustration that can occur with long questionnaires, and as a result, ensure that the answers given are as accurate as possible. The SUS questions were slightly modified to replace the word 'system' with the word 'software' to relate more accurately to the study at hand. Additionally, the SUS was augmented with five extra questions, also rated on a Likert scale. These extra questions were designed to assess the effectiveness of the documentation, and the results were reviewed separately from the main SUS score. Completion of the questionnaire was immediately followed by short semi-structured interviews to gain more in depth information.

Participants were asked to repeat a shorter version of the test after a period of one week, this time without the chance to refer to the documentation. This test was to investigate the learnability of the software. The study used usability metrics divided into three categories to measure the usability of Polaris. These three categories were chosen in accordance with the international standard ISO 9241-11 [12], which defines usability as comprising of effectiveness (the ability of users to complete tasks and goals), efficiency (the level of resources consumed in performing tasks), and satisfaction (a user's subjective reactions to using the system).

Using this definition, the usability metrics used are shown below:

Effectiveness was measured by

- The number of references users make to Polaris documentation;
- The length of time spent referring to documentation;
- The number of users who remembered how to complete goals after a period of inactivity;
- The number of errors encountered.

Efficiency was measured by

- The time taken to complete each task;
- The number of mouse clicks taken to complete each tasks.

Satisfaction was assessed through

- Questionnaires and short semi-structured interviews to gather subjective data.

A summary of the collected quantitative data is presented in table 2.

Table 2. Summary of quantitative data collected

Task #	Number of participants who successfully completed the task (out of ten)	Average time taken to complete the task (minutes:seconds)	Average number of documentation look ups	Average time spent for each documentation lookup (minutes:seconds)	Average number# of errors encountered (cumulative)	Average number of mouse clicks
1	7	11:56	5	1:41	4	31
2	10	4:40	2	1:01	0	15
3	10	5:17	1	0:16	2	12
4	10	4:32	0	0:00	0	14
5	4	15:22	15	0:20	25	56
6	4	4:51	0	0:16	15	14
7	10	6:05	2	0:40	6	30
8	10	3:00	1	0:32	7	12

5. WHAT WE LEARNED

Polaris is different from traditional anti-virus software in that once installed it does not require updates. Once the one-off polarization procedure is completed, the virus protection is integrated into the application itself, and the protection offered is independent of virus versions. This would seem to instantly offer greater usability, however, as we will see later this effect may have been negated by the burden on the user to make other security-related decisions.

After the tests, all participants showed an understanding of what Polaris was trying to achieve, but only two participants understood the idea of having multiple pets for a single applications. The designers of Polaris wanted total transparency, but participants in this study did notice its presence. Polaris required initial configuration, required users to make decisions as to how to open files safely, and produced error messages. It is possible that the transparency may increase over time, as the users configure it to suit their needs, but as this study took place over a short time scale, this cannot be determined.

Several usability problems were discovered, which are discussed below, under each of the three metric categories used. This is followed by brief discussions on decision making and user apathy.

5.1 Effectiveness

Some participants had difficulty in identifying whether the application they were using had been polarized or not. This was made evident in task 1 when three out of the ten participants were unable to discern between polarized and normal applications. A further two participants were unable to discern when tested in Windows XP service pack 1, but were able to correctly identify the difference in XP service pack 2. This is due to a bug in the software which prevented the visual differentiation from working in service pack 1. The remaining five participants were unable to identify the difference immediately, but had to take long measures such as examining the list of polarized applications in Polaris. Many participants commented in the interviews that the visual difference between normal and polarized applications was not apparent enough. The participants who could not identify whether applications had been polarized suffered from further ramifications in later tests, because they assuming they were being protected by Polaris, when in fact they were not.

The number of references to the documentation, and number of errors encountered were distributed fairly equally across all the tasks but one. In task 5 users were asked to customise the toolbar of Microsoft outlook so that it included options to use Polaris on e-mail attachments. Instructions on how to do this were in the documentation, however only four out of ten users ultimately managed to complete this task. This task required referring to the documentation five times as often as was the average for all other tasks, and produced three times more errors. This demonstrates the problems encountered when users are asked to set up software themselves. The Polaris development team is making efforts to automate this process as much as possible for the beta release.

The average length of time spent referring to the documentation for the first task was eight minutes 25 seconds, as users familiarised themselves with the software. During subsequent tasks users looked at the documentation rapidly for brief periods of around 15 seconds, which shows their need for detailed guidance when first using the software.

Polaris displayed error messages sometimes with no apparent cause, and frequently with no explanation of how to resolve the error. The participants were seen to quickly dismiss these error messages, especially if they had already seen the same error at least once. For example, the following error was given when trying to open a downloaded application using the 'Icebox' feature of Polaris:

Application has generated an exception that could not be handled.

Process id=0xf38 (3896), Thread id=0fx3c (3900)

Click OK to terminate the application.
Click CANCEL to debug the application.

This could have been better communicated to the user; the meaning of this error message is as follows:

This application cannot be opened in the Icebox. Try Polarizing it, or if it is from a trusted source, open it without Polaris.

The participants completed a shortened version of the test after one week in order to test the learnability of software. This time there no documentation was made available. This test had mixed results, with some users being able to quickly complete tasks that others could not remember how to do, and vice-versa. Some users commented that it would be easier if there were a context sensitive menu from which they could choose several Polaris options when right clicking files and hyperlinks. Polarizing an application was a task that was widely successful, it is thought that this is due to the interface being simple and intuitive (select an application and click 'Polarize'). However trying out an application downloaded from the Internet was a task with a low success rate. This is a task that required the participant to perform actions above and beyond what is normally required to run a downloaded application. These actions are not obvious, must be repeated very frequently, take extra time and hold no apparent advantage for the user, as they can run an application (albeit unsafely) without any additional steps. In these situations the participants chose to run the application in the normal way rather than use Polaris. Learning and using Polaris presents barriers to getting work done quickly, the benefits of which hold too little value for the participants to put in the extra effort. Participants put emphasis on the speed of doing things and did not like to be slowed down. The participants' apathy towards security and willingness to compromise security is further discussed in section 5.5.

5.2 Efficiency

Task 5, that required customizing the Outlook toolbar, took significantly longer (up to 15 minutes) and required up to four times as many mouse clicks as the other tasks. Users struggled with this task and exerted more effort than with other tasks. This task required much more customisation of the software than any other. We believe placing this burden on the user decreased the usability of the software. All other tasks required an average of between 12 and 30 mouse clicks, and took on average between three minutes and six minutes to complete, except task one which took 11:56, due to the initial reading of the documentation. This seems quite reasonable considering the users had never used the software before.

5.3 Satisfaction

The SUS scale gave a mean average score of 44.2 out of 100. Most users indicated that the software was cumbersome to use, and they would not like to use it frequently. The participants showed frustration at nonsensical error messages, and thought that the various features of Polaris were not well enough integrated. Some participants commented that more context sensitive menus would make Polaris easier to use.

One participant assumed that Polaris was automatically protecting their files at all times, when in fact some of the applications they were using were not under the protection of Polaris. This user had a high expectation of the security software in that they didn't expect to have to take any explicit action in order to be protected.

5.4 Decision Making

The most serious usability problems arose when a considerable responsibility in decision making was passed onto the user. The most noticeable instance of this was when participants were expected to polarize an application multiple times for different uses. The Polaris documentation states:

"Each Pet has permission to read and write any files opened by that Pet. So, if you've opened one spreadsheet received as spam and another spreadsheet containing critical information, a virus running in the spam spreadsheet could destroy the information in the critical file. In order to prevent this attack, you may create more than one Pet for the same application"

It should be noted that malicious files opened in pets only present a security risk to other documents that are open in the same pet. Polaris provides protection over the system area of the registry, and the Windows directory, which are often targets for attack.

The participants were presented with a scenario to test their use of multiple pets. They were given several hyperlinks to open in a web browser. One was a secure Internet banking site they had to log into, and the others were unknown sites on publicly editable domains, which were engineered to appear untrustworthy.

When interviewed, just six out of the 13 total participants claimed they knew that it was possible to create multiple pets for one application, any only two of these knew why this would be desirable.

One of the participants who knew why multiple pets might be desirable created a pet browser to log into a secure internet banking website, and after using the site, indicated that he thought it was secure, safe, and trustworthy. He was then sent two unknown hyperlinks via e-mail, which he believed to be insecure, unsafe, and not trustworthy. He was aware that any malicious code from the distrusted site may be able to affect information from the secure banking session, but despite all of this, he still did not create multiple pet browsers. Instead, he opened the un-trusted links in the same browser pet as was used for the secure banking session, thus knowingly compromising the security offered by Polaris.

In fact, none of the participants used multiple browser pets.

5.5 Apathy to Security

When asked to download an application from a website and try it out securely, most participants considered the goal here was opening the application, rather than protecting their security. As such, nine out of 11 (82%) of the participants (this includes one pilot participant-the other two pilots were discounted due to technical difficulties) simply double clicked the application and opened it without Polaris, compromising the security of their PC. Some of these then went on to use Polaris to protect their security, but by this point the damage could have already been done, had the application been malicious.

During the experiment the participants were asked to judge the safety, security, and trustworthiness of the hyperlinks before visiting them, and in the interview they were asked to do the same after having visited the sites. The results showed that they were all able to distinguish between sites that should and should not be trusted. They based their decisions on previous experience, the appearance of the e-mail that contained the hyperlink, the reputation of the web sites (e.g. Yahoo), and by identifying the

padlock symbol in the browser for the secure site. Although the participants had a high awareness of the security risks of the Internet, and knew the possible consequences of their actions, they were not any more protective of the PC's security, in fact they showed total apathy towards the protection of files, and knowingly compromised their security.

The apathy encountered during the tests seems to be due to the users' persistent attitudes towards security. When questioned, the two users who did know the purpose of creating multiple pets did not put their theories into practice because they simply did not care about the consequences. Some participants also indicated that their data was not important to anyone but themselves, and therefore not worth taking effort to protect. Participants also indicated that completing the task at hand was more important than protecting their security and it was observed on several occasions that they would try to use Polaris, but if they were unsuccessful in their first attempt they would bypass it to open files without protection. The experimental conditions in which the participants were observed may have affected their behavior, and would agree with the data from Weirich & Sasse [20], which showed that users will not make good security decisions unless they believe they are at risk. In any case, given that users will knowingly compromise their PC security, we believe it is unreasonable to expect them to make continual security related decisions, such as when to use a different browser pet, in everyday life.

If the user set up the Polaris software, but subsequently did not use it properly, as was observed in these tests, their level of security would be comparable to ordinary Windows users. An exception to this would be the cases where users believed they were being protected by Polaris when in fact they were not; this may lead to complacency over security and increased risk of attack. If, however, Polaris is imposed on the user, for example by a corporate security policy, they would have to work through the usability difficulties outlined in this report. These include confusion over when protection is being offered by Polaris, annoying error messages, difficulty in customizing the software to work with e-mail clients, and the inability (or lack of motivation) to decide when to use multiple pets for a single application. These problems would hinder the user in their work and may render the protection offered by Polaris ineffective.

The visual distinction between polarized and non-polarized windows needs to be much stronger, as users are likely to have a large number of applications in a mixed state of polarization, and need to know immediately and intuitively whether they are being protected or not. Polaris should be more tightly integrated with the operating system so that context sensitive menus can be used. The need to have a separate pet for each trust category seems an impassable problem for the average user, and one which is inherent to the application of the POLA principle. As such, the solution to this problem requires more thought then the simple interface changes which can remedy other difficulties. Perhaps each instance of an application pet could store its temporary data in a separate disk area which is cleared after the instance is closed. This would remove the risk of different application instances interfering with one another's data, and remove the need for the user to make continual trust decisions, but at the expense of not allowing long-lasting data such as cookies to be stored and used.

6. CONCLUSIONS

This study has found that there are usability problems even in a product that was designed to reduce them. These problems can be attributed to the fact that the operation of the software was not as transparent as its designers had hoped it to be. As [6] suggests, participants did not know when or how to make security related decisions, so these usability problems may be alleviated by removing the decision making responsibility from the user, thus making the software more transparent. However, care should be taken to only remove this power from the user where the system can do a better job (make better decisions). Removing control from the user at times when only they can effectively decide when and how to share information can become problematic [4].

The participants' willingness to compromise security was a worrying discovery. They rationalized this behavior by declaring the speed and ease with which tasks were completed to be more important than the protection of their files. The participants prefer speed to security, so this apathy may be counteracted by ensuring that the secure way of doing things is the fastest way. It would also be valuable to increase users' sense of worth for their data and increase their motivation for protecting it. It may at first seem that education is the best way of achieving this, but previous research has shown that education was rarely effective for such matters. Instead, perhaps a visual indicator which shows users the level of risk their data is under can be used. A similar function is provided by the upcoming Internet Explorer 7, in which the address bar changes from green to amber to red depending on the authenticity of the website. In a similar vein, the eBay Toolbar [7] alerts users when they are about to submit their password to an unverified web site. It is likely that over time these alerts will annoy users and they will learn to quickly ignore them, but the notion of making users more aware of their actions is commendable.

Polaris could be made faster by having pets automatically created for programs upon installation. Furthermore, if each pet uses a separate temporary disk area to store information, which is cleared after the pet is closed, this could prevent the user from having to make decisions as to when to use multiple pets, at the expense of not facilitating permanent data such as cookies and cache files to be used.

The study also corroborated the notion that users quickly dismiss confusing error messages [11, 25]. They see the message as a hindrance rather than a help, and their habit of clicking away messages before reading them raises doubt that message boxes are an effective way of alerting the user to an event.

The goal of making Windows security more usable, whilst admirable, seems unlikely to be successful since it is a post-hoc consideration. This strengthens the argument made by other HCI-SEC researchers [e.g. 1, 8, 24], that security and usability must be developed in unison from concept right through to development as an integral part of the system if they are ever to align perfectly. But even if considered at this early stage, the tradeoffs between security and usability make for difficult design decisions, and since it seems that no products have so far been designed in this way, it is not possible to evaluate the efficacy of this method.

7. FUTURE WORK

More research is required to assess how to simplify and automate complex security software, in order that fewer burdens are placed

on the user to make constant security related decisions, and to discover which decision making points can be safely eliminated.

It is not known whether the problems associated with decision making found in this study are also to be found with other types of security software, nor what impact will be made on the user's satisfaction if the decision making responsibility is completely removed from them. These problems also require further investigations.

A limitation of this study is that the participants were aware that they were in experimental conditions, and as such were under no real risk from attack. This may have affected their motivation to protect the PC they were using. A repeat of the study in which participants could be induced to have a high motivation for protecting the files as if they were their own would help address this issue.

8. ACKNOWLEDGMENTS

We would like to thank everybody involved with the Polaris project at Hewlett-Packard Laboratories, Palo Alto, CA, USA, who were very helpful, and gave us the opportunity to study the software at an early stage.

9. REFERENCES

[1] Balfanz, D., D.K. Smetters, and R.E. Grinter, *In search of usable security: Five lessons from the field.* IEEE security and privacy, 2004. **2**(5): p. 19-24.

[2] Boehm, B.W., *A spiral model of software development and enhancement.* Computer, 1988. **21**(5): p. 61-72.

[3] Brooke, J., *Sus: A quick and dirty usability scale*, in *Usability evaluation in industry*, P. Jordan, B. Thomas, and B. Weerdmeester, Editors. 1996, Taylor and Francis: London.

[4] de Paula, R., X. Ding, P. Dourish, K. Nies, B. Pillet, D. Redmiles, J. Ren, J. Rode, and R. Silva Filho. *Two experiences designing for effective security*. in *Symposium On Usable Privacy and Security*. 25-34 2005. Pittsburgh, Pennsylvania: ACM Press.

[5] Dourish, P., J. Delgado de la Flor, and M. Joseph. *Security as a practical problem: Some preliminary observations of everyday mental models*. in *Workshop on HCI and Security Systems, CHI2003*2003. Fort Lauderdale, Florida, USA: ACM.

[6] Dourish , P. and D. Redmiles. *An approach to usable security based on event monitoring and visualization*. in *2002 Workshop on new security paradigms*. 75-81 2002. Virginia Beach, Virginia: ACM Press.

[7] eBay, Using ebay toolbar's account guard, http://pages.ebay.com/help/confidence/account-guard.html. 2006. Accessed on 20th April 2006.

[8] Flechais, I., A.M. Sasse, and S.M.V. Hailes. *Bringing security home: A process for developing secure and usable systems*. in *Workshop on new security paradigms*. 49-57 2003. Ascona, Switzerland: ACM Press.

[9] Garfinkel, S.L. and R.C. Miller. *Johnny 2: A user test of key continuity management with s/mime and outlook express*. in *Symposium On Usable Privacy and Security*. 13-24 2005. Pittsburgh, Pennsylvania, USA: ACM Press.

[10] Gerd, D. and T. Markotten. *User-centered security engineering*. in *Nordu2002*2002. Helsinki, Finland.

[11] Gutmann, P., *Inadvertent case study in ssl server cert effectiveness*, hcisec@Yahoogroups.com, Editor. 2005.

[12] ISO, *Ergonomic requirements for office work with visual display terminals (vdts) - part 11: Guidance on usability*. 1998, BSI.

[13] Nielsen, J., Security & human factors, http://www.useit.com/alertbox/20001126.html. 2000. Accessed on 20th February 2006.

[14] Nielsen, J., Why you only need to test with 5 users, http://www.useit.com/alertbox/20000319.html. 2000. Accessed on 29th November 2005.

[15] Roth, V., K. Richter, and R. Freidinger. *A pin-entry method resilient against shoulder surfing*. in *Proceedings of the 11th ACM conference on computer and communications security*. 236 - 245 2004. Washington DC, USA: ACM Press.

[16] Saltzer, J. and M. Schroeder, *The protection of information in computer systems.* Proceedings of the IEEE, 1975. **63**(9): p. 1278-1308.

[17] Stiegler, M., A.H. Karp, K.-P. Yee, and M. Miller, Polaris: Virus safe computing for windows xp. 2004, HP. http://www.hpl.hp.com/techreports/2004/HPL-2004-221.html

[18] Straub, T. and H. Baier. *A framework for evaluating the usability and the utility of pki-enabled applications*. in *EuroPKI*. 112-125 2004. Samos Island, Greece: Springer-Verlag GmbH.

[19] Virzi, R.A., *Refining the test phase of usability evaluation: How many subjects is enough?* Human factors, 1992. **34**(4): p. 457-468.

[20] Weirich, D. and M.A. Sasse. *Pretty good persuasion: A first step towards effective password security for the real world*. in *New Security Paradigms Workshop*. 137-143 2001. Cloudcroft, NM, USA: ACM Press.

[21] Whitten, A. and J.D. Tygar. *Why johnny can't encrypt: A usability evaluation of pgp 5.0*. in *Proceedings of the 8th USENIX security symposium*. 169–184 1999. Washington, D.C.

[22] Yan, J., A. Blackwell, R. Anderson, and A. Grant, *Password memorability and security: Empirical results.* IEEE security and privacy, 2005. **2**(5): p. 25-30.

[23] Yee, K.-P. *User interaction design for secure systems*. in *4th international conference on information and communications security*. 278–290 2002. Singapore: Springer Verlag.

[24] Yee, K.-P., *Aligning security and usability.* IEEE security and privacy, 2004. **2**(5): p. 48-55.

[25] Zurko, M.E., C. Kaufman, K. Spanbauer, and C. Bassett. *Did you ever have to make up your mind? What notes users do when faced with a security decision.* in *Computer Security Applications Conference*. 371 - 381 2002. Las Vegas, Nevada, USA: IEEE.

[26] Zurko, M.E. and R.T. Simon. *User-centered security*. in *Proceedings of the 6th new security paradigms workshop*. 27-33 1996. CA: ACM.