Understanding the weaknesses of human-protocol interaction

Marcelo Carlomagno Carlos* and Geraint Price

Royal Holloway University of London,
Egham, Surrey, TW20 0EX, United Kingdom
{marcelo.carlos.2009,geraint.price}@rhul.ac.uk

Abstract. A significant number of attacks on systems are against the non-cryptographic components such as the human interaction with the system. In this paper, we propose a taxonomy of human-protocol interaction weaknesses. This set of weaknesses presents a harmonization of many findings from different research areas. In doing so we collate the most common human-interaction problems that can potentially result in successful attacks against protocol implementations. We then map these weaknesses onto a set of design recommendations aimed to minimize those weaknesses.

Key words: human-protocol interaction, human factors, usable security

1 Introduction

Sometimes, even some of the most secure and robust protocols are vulnerable to attacks when implemented. The reason for this is that a significant number of these attacks are against the non-cryptographic components, such as the human-protocol interaction.

In protocol design and analysis, the human interaction is usually part of the assumptions and not specifically included in the description. However, user behaviour is often unpredictable, making the assumptions not precise enough [11, 16]. Despite that unpredictable nature, there are some common design errors and weak-assumptions that can be avoided if previously known.

We conducted a thorough study of the existing work, among different areas, to learn and understand the most common characteristics and behavioural patterns of human-protocol interaction. Based on the results of this study, we propose in this paper a unified set of human-protocol weaknesses that merges the findings from different research areas into a harmonised taxonomy. In particular, we focus on human characteristics that are usually overlooked during the protocol design process. Based on this set, we then discuss ways we can tackle these weaknesses and minimize their impacts. Our analysis is partly based on related research findings, but is also based on our own proposals which evolved from our

* Supported by CNPq/Brazil

taxonomy of weaknesses. Ultimately, this allows us to also present a set of design recommendations to address the problems inherent in the human-protocol interaction. What is interesting is that this set is not simply a linear evolution of the taxonomy of weaknesses. What we can see is that a second independent layer of structure emerges when we separately consider the categorisation of solutions to the problems we collate and identify in the taxonomy of weaknesses.

In Section 2 we present and discuss related work. In Section 3 we present our proposed set of overlooked components of human-protocol interaction. Based on the that set, we discuss, in Section 4, ways to minimize the impact of those weaknesses. In Section 5 we present the design recommendations derived from the discussion in the previous section. Our conclusions and some opportunities for further research are discussed in Section 6.

2 Overview

Human computer interaction is a topic which spans several different areas, including computer science, sociology and psychology. A large portion of research in this subject is related to design and usability of security systems. Each of these research areas independently address different layers of systems security.

Figure 1 gives us an overview of the layers involved. The specification layer, represents the protocol specification; the application layer contains the implementation of the specified protocol in an application; the interface brings a point of interaction between the application and user layers; finally, the user layer represents a user of the application.

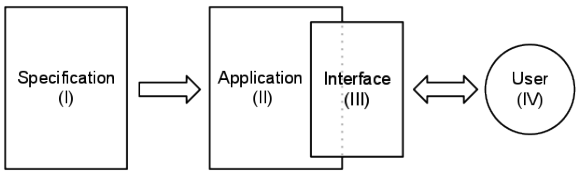

Fig. 1. Human-protocol interaction layers

Software engineering is focused on the application's design and implementation (application layer); HCISec focus on the interface and usability aspects (interface and user layers); computer security design focuses on the design and implementation level (specification and application). As we can see, most of research focus on a specific layer rather than the whole set. This creates a gap specially between the specification and implementation layers where, the human-protocol interaction involved, has received little attention. The human-protocol perspective, which we focus in this paper, goes trough all the layers, such as specification, application, interface and user.

Within the scope of human-protocol interaction we consider any type of action performed by a user that might impact on the security properties of a system. Since the interaction is usually made via a software interface, we have to consider usability issues. Additionally, we need to look at the implementation and

specification levels. In a protocol specification, the human-protocol interaction is usually part of the design assumptions, that is, static components where actions are assumed to happen without being explicitly included into the specification. When implemented, these static choices are then replaced by dynamic user-interactions. It is often the case where the assumptions are too strong, that is, it is very unlikely for a implementation to provide the expected security properties. Our work focuses on detecting patterns of problems that occur during the human-protocol interaction and highlight human characteristics that are often overlooked during the protocol design and implementation. We explore the findings of research within these layers to construct a broader point of view.

As mentioned above, different research fields address security issues in different ways. However, we were able to observe overlaps in the definitions. These overlaps occur due to the use of different terminologies and are also due to the distinct research goals. For example, there have been studies of phishing attacks [14, 21, 29, 7, 6]; users' susceptibility to attacks [9, 24]; factors exploited to allow an attack to be successful [27, 8, 12, 25] and others. Among this wealth of studies, we found similar findings labelled in different ways. We also found that some findings applied to a specific context, which can be extended to different contexts. Each of these existing areas of research independently contributed to the construction of the set of human-protocol interaction weaknesses and recommendations we propose in this document.

3 Frequently overlooked components of human-protocol interaction

The interaction between computers is relatively easy to define and the results are, hopefully, predictable. However, defining or predicting human behaviour is a challenging task. It requires a more subtle approach, commonly based on empirical results [16]. Despite this complicating factor, we can create a generic (but not perfect) human-protocol interaction model by using empirical and statistical information and use it to improve the human-protocol interaction process.

Therefore, it is necessary to study and analyse the most common characteristics and behavioural patterns regarding human-protocol interaction. There is a significant amount of research which maps human characteristics (or principles, weaknesses, etc). This existing body of work offers important and relevant insights that constitute the basis of the set of overlooked components of human-protocol interaction we discuss during this section. By analysing existing work and detecting the common components among them, we could define our list of five main overlooked components of human-protocol interaction: user knowledge, authentication capabilities, decision making influencing factors, bounded attention, and inherent characteristics.

3.1 User knowledge

We define knowledge as familiarity, awareness, experience or understanding of a certain subject. Within the human-protocol interaction context, users' knowl-

edge certainly is an important factor to be analysed. We have seen several situations where this factor (or the lack of it) is exploited by attackers. Therefore, a secure human-protocol interaction should carefully deal with users' knowledge.

Phishing attacks provide a very interesting case study from where we can evaluate human-protocol interaction. The main reason is that, in many cases, the user has to interact with an implementation of the SSL/TLS protocol. Many studies and experiments [8, 29, 9, 24, 14] have shown that, indeed, users are not familiar with computer systems, security, security indicators and risks (such as web frauds). When an attacker is able to perceive a weakness in the victim's level of knowledge, it is relatively easy to manipulate and exploit a users' lack of knowledge to successfully attack a protocol. In summary, we can list the knowledge-related issues that are most commonly exploited by attackers:

Lack of knowledge of computing – many people do not have proper understanding of how operating systems, networks and protocols work [8, 9, 29].

Lack of knowledge of security – users do not have knowledge about digital certificates, cryptography and most of security technologies [8, 29, 14, 9].

Lack of knowledge of security threats – many users do not know they can be attacked; that spoofing websites is possible; and what the techniques used by the attackers are [8, 29].

Inaccurate mental models – people frequently construct their own concepts about computing, security and threats. It is often the case these concepts are not correct [1, 24].

Human-protocol interaction cannot rely on users' computing/security knowledge and awareness. A secure human-protocol interaction must consider user's knowledge and, preferably not require higher training levels.

3.2 Authentication capabilities

An authentication performed by a user is a task where the user verifies that the authenticating party is whom it is expected to be. People frequently make use of visual cues as an important authentication tool. However, there are studies [26, 27, 8, 29, 14] that show that this visual authentication mechanism is weak and unreliable in some cases. In these specific situations human authentication capabilities should not be used as an important component in the protocol specification. In general, we found four different users' authentication skills that are well known by attackers, but not always correctly addressed by designers:

Users are good at authenticating people they know – in general, users are very good and efficient at authenticating people they know [26].

Users are not good at authenticating objects – usually, users have problems in authenticating objects [26, 6, 29]. Objects are easy to spoof and when facing a spoofed object, it is likely that a user will perceive it as original.

Users are not good at authenticating strangers – people are not good at establishing whether someone belongs to a designated class (e.g. policeman)

[26]. When authenticating strangers, users have to make use of other authenticating factors, such as documents or references given by someone else (e.g. physical attributes). This shifts the authentication type to object-based, which it is not precise enough to be used in security protocols.

Users are not good at authenticating digital objects – in the same way as real objects, digital objects, such as websites, software and email are also not easily authenticated by users. By creating visually identical (or very similar) copies of the original source, attackers can fool users into believing they are contacting the entity they trust [9, 14, 8].

Directly or indirectly, most scams exploit the false acceptance of a spoofed content by users and almost all scams are forms of deception [27]. Asking a user to authenticate an object (e.g. an online banking website) by checking its elements (e.g. digital certificates, padlocks, etc) does not represent a proper translation from the authentication design goal to its implementation. In fact, it is very likely that the authentication task, despite technically feasible, will not be performed properly, introducing a security breach.

3.3 Decision making influencing factors

There are different factors that need to be taken into account when considering users' decision making and its influencing factors. These aspects include personal and environmental issues. Despite the differences, the core concept behind this component is that users can be influenced to make different (and potentially damaging) decisions to those they would usually make. Thus, when designing human-protocol interaction, security engineers must be aware of which factors may influence the user's decisions and check whether this decision under influence can introduce security breaches or not.

The most common influencing factors found are:

Social conditioning – when people receive commands from strangers, they are unlikely to follow that command without questioning the request. However, when the command comes from a recognized authority (or someone mimicking an authority), people are very likely to obey this command. This happens because people are trained to accept commands from certain people, such as police officers, without further rationalization [27, 24, 17].

User's principles – victims' principles such as need, greed or dishonesty, make them vulnerable because the attacker can use them to force the victim to behave in a predictable manner [27].

Time constraints – the main idea behind this factor is to push the victim to make a decision without sufficient time to rationalize the decision. Consequently, the actions taken by the victim tend to be more predictable and easier to manipulate [27]. The decision strategies used under time pressure is typically based on affective and intuitive heuristics, rather than on a reasoned examination of all the possible options [12].

Shared risk – this aspect is found in real-world situations where someone accepts a risk because there are many others sharing the same risk [27].

Fear – many techniques such as scareware are effectively used by attackers to scare people and make them fall into attacks (e.g. Mac defender malware[1]).

Users' decision making factors involves many different factors that should be carefully analysed. Even trained users might have their decision strategies shifted under certain circumstances. People will make errors and will eventually make wrong decisions. It is important to identify potential situations where this component might be exploited and make the system insensitive to them.

3.4 Bounded Attention

Users are focused on their main task, and consequently, most of their attention is bound to the activity of performing that task. Security protocols are frequently used as part of a computational system or software. Consequently, from the users' perspective, the protocol used and its security aspects are a secondary concern. As a result users have a tendency to notice only what they are interested in and do not pay attention the fact those security mechanisms were created to protect them from attacks [27, 8]. In our research, we found four main factors which can potentially weaken the security aspects of the human-protocol interaction:

Lack of attention to security – user's focus is not on the security aspects of the system. Consequently, security checks are executed less carefully [8, 9].

Lack of attention to the absence of security – In the same way that security checks may be dismissed without further rationalization, their absence might not be noticed by the users [8, 28].

Security in a secondary workflow – users are more likely to finish their main task rather than stop it due to a security warning. Security checks that occur outside of the main task interrupt the user's focus and are more likely to be dismissed without much consideration. Users will try to finish their main tasks if they believe they are more important then the security tasks, even if there are potential risks [29, 5].

Conditioning – an excessive number security interruptions ends up training users to dismiss warnings, pop-up boxes and any other security interruptions in a insecure way because this is the only (or the simplest) way to finish their tasks [2]. A excessive number of warnings, in the course of time, can make users become less inclined to take them seriously in the future [19, 5].

As we can see, security should be included in the main workflow to be effective. Simply warning users by stating that something is wrong is not sufficient: they need to be provided with a safe alternative to achieve their goals [29]. Bounded attention may also be affected by user's knowledge. For example, a user may not know what security cues they should look for or whether the operation being performed is insecure or not.

[1] http://support.apple.com/kb/ht4650

3.5 Inherent characteristics

Human skills is a broad concept. It includes proficiency or ability that is acquired or developed through training or experience. Overlooked inherent characteristics encompasses situations where human skills might not be enough to perform an activity or task as intended. It also includes particularities of human behaviour. We cannot expect that humans behave similarly to a computer, nor believe they share similar skills. By equating these two different components during the human-protocol interaction, a series of security threats may arise. A usable and secure design must consider human abilities and check what people can, and more importantly, what they cannot do well [7]. There are several skills we should consider when designing secure systems:

Memory limitations – human capacity for working memory is limited and decays over time [24]. We cannot expect users to remember large and random keys nor recall dozens or hundreds of different passwords [24, 5]. People also cannot "forget on demand". So, even undesired items will remain in memory even when they are no longer needed [24].

Lapses and Slips – lapses and slips errors occur when the plan to achieve a certain goal is correct, but an error happen when a required action is forgotten (e.g. a step in a sequence of actions) or an action performed incorrectly (e.g. pressing the wrong button) respectively [22, 5].

Problem solving limitations – some problems can be easily solved by some users but the same problem can be a complex task for others. This limitation can be influenced by many of the previously presented weaknesses, such as lack of knowledge or bounded attention.

Task termination – when users finish their main task, they might leave the subsidiary tasks incomplete. For example, a user accessing a webmail, may leave the computer without logging out. This is ok on a private computer, but is it a security threat in public environments. In a similar manner, users may terminate the interaction if they assume there is no alternative to proceed due to a fault or an unexpected system state [23].

Non-deterministic behaviour – As opposed to the previous limitations, this issue is not related to a limited set of capabilities, in fact it is the opposite situation. According to Ruksenas [23], in any situation, any one of several cognitively plausible behaviours might be taken.

We cannot expect that users have skills or abilities they do not have. The human-protocol interaction should be designed considering the human's inherent characteristics and checking whether the task given to the user is feasible or not.

4 Minimizing the weaknesses in the interaction

After merging several research findings into a harmonized and limited set of often overlooked human-protocol interaction components, a set of design recommendations to minimize the effects of these weaknesses is a clear next step. Despite

the wide range of users' characteristics and behavioural patterns, we proposed a set overlooked components of human-protocol interaction, and from them, we could develop a set of design recommendations.

To construct a set of recommendations on how to reduce the impact of these weaknesses, we initially attempted to make a one-to-one association between a weakness and a corresponding recommendation, where, for each weakness, we proposed a design recommendation. We independently analysed each of the influencing factors on the human-protocol interaction weaknesses, and, by making use of our findings and the results of related work we proposed design recommendations for each influencing factor.

What we found was that, for some factors, even those belonging to the same category of weakness, have to be treated in different ways. However, the opposite situation was also found, when factors from different weaknesses could be handled in a similar manner.

Figure 2 represents the associations between our set of frequently overlooked components of human-protocol interaction and our design recommendations. As we can see, almost all of components are linked to two or more recommendations. This happens due to the internal subdivision of each component (described in Section 3). In many cases, each sub-component had to be treated in a different way. Due to space constraints, the description of how each subcomponent is mapped to a design recommendation were left out of this paper.

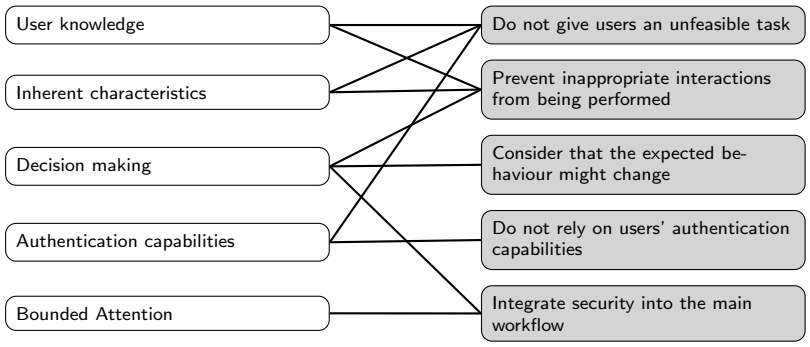

Fig. 2. Mapping human weaknesses into design recommendations

5 Design recommendations

By analysing the human-protocol interaction weakness we presented in section 3 and by exploring related research findings, we were able to discuss ways to tackle these weaknesses and minimize their impacts. The associations among the weaknesses and solutions to minimize their impacts were a key factor that allowed us to identify the design recommendations we discuss in this section. By proposing a set of design recommendations, we introduce guidelines to help designers to overcome the problems presented in Section 3. Due to space constraints, some examples to illustrate how each recommendation can be applied in real word scenarios were left out of this paper.

5.1 Do not give users an unfeasible task

Humans have different levels of knowledge in a wide range of areas. Some people have stronger abilities in subjects such as logic or mathematics and others are better dealing with human sciences and so on. Some protocols might be expected to be used only by specialists and consequently require a higher level of knowledge. However, there are other protocols that are designed for general purpose use and, consequently, used by people that have different levels and areas of knowledge. In both cases, protocols should be designed and implemented keeping in mind the level of knowledge of the person who will interact with it.

An example of a task that depending on the context may be considered unfeasible, is to require the user to generate input data to be used as part of the protocol workflow (e.g. random passwords). Certain inputs can represent fundamental elements of the whole system security and should be carefully analysed. In the Kerberos [18] protocol, a user input (password) represents a fundamental part of the protocol security. If we simply allow the user to create a password, a weak password might be generated and consequently compromise the protocol security [3]. On the other hand, defining password policies (e.g. minimum length) can also generate other types of problems such as information disclosure [13]. Thus, it is necessary to find alternatives to produce input data to a protocol which has sufficient quality to be used as a trustworthy source, as well as make its generation and use feasible to ordinary humans.

The recommendations about not giving users an unfeasible task can be summarized in the following list:

- Identify where the security conditions of the protocol relies on a task performed by users and identify the level of knowledge and skills of the target audience.
- Check whether the task requires specific types of knowledge or skills. If it does require, check whether the target audience attend possesses prerequisites. The more generic the audience, the lower level of understanding and skills should be required.
- Avoid using user input as a main part of the establishment of security properties of the protocol. If a user input is required (e.g. user password used as the seed to a key) and check if it is necessary to create policies to guarantee the good quality of the input.

5.2 Do not rely on users' authentication capabilities

People are very good at recognizing people they already know, but they are not good when authenticating strangers or objects [27]. Thus, except for particular cases, such as human-human interaction between people that know each other, we cannot rely on human's authentication capabilities and consequently should not include this task in the human-protocol interaction.

During the SSL/TLS protocol handshake, when an unknown server certificate is presented to the client's browser, users are asked whether they trust that

certificate or not. By being asked that question, users are receiving an authentication task. However, asking users to authenticate an object (a digital certificate in this case) is not recommended because humans are not capable of authenticating digital objects properly, and therefore, this authentication process becomes insecure. In this specific cause, user's knowledge is also not properly considered, since this authentication task a high level of knowledge.

The designer, to avoid security failures due to authentication mistakes, should:

- identify where the security properties of a protocol relies on an authentication task performed by humans [27].
- check whether the authentication task includes authenticating unknown people or objects.
- verify if the authentication task given to the user is feasible for a ordinary verifier, not requiring specific technical knowledge [27].

5.3 Integrate security into the main workflow

When security is a secondary activity for the user, it tends to be ignored or underrated. Warnings, messages and prompts asking users whether to accept a certain change in the security context tend to be ignored by users, compromising the protocol security [25, 29]. Moreover, most current implementations are plug-ins or amendments to existing designs which are attempts to overcome inherited design problems. Security concerns about human-protocol interaction should be part of the design and included into the main path of the protocol's flow.

The following recommendations summarize some considerations that should be used during the protocol design:

- If a decision is critical to the security of the protocol, integrate the security concerns into the critical path of their tasks. By doing it, users will be forced to interact with it, and will not be able to ignore it [29].
- Use active interruption other than passive warnings. However, consider the usability impact of the new design to avoid an excessive use of warnings, which may reduce the attention given to the them over time [29].
- Incorporate security decisions into the users' workflow, and, whenever possible, infer authorization from acts that are already part of their primary task [30].
- Respect user intentions. Warning users that something is wrong and advising them not to proceed (while still giving them the option to continue) is not the right approach [29].

In the SSL/TLS protocol implementation, we could apply these recommendations by changing the message presented to the user regarding the untrusted server certificate. Currently a message from the browser to the user is sent via an active warning (a window asking whether the user wants to accept the certificate). Despite some recent changes in the implementation of these warnings (which made them more effective [10]) we still believe that once users learn how to dismiss these warnings, the efficiency of the this type of warning tend to be

reduced. Thus, a third warning type might be needed. We call "interactive warning" a new type of warning that instead of informing or interrupting users, it makes the user interact with the protocol.

In the SSL/TLS certificate warning, an interactive warning could, for example, be implemented by asking the user to input a web address confirmation. Consequently, the user would only be allowed to access the website if the "Common Name" field in the server's certificate matches the address typed by the user. By making this change, attacks that exploits users' bounded attention, for example, would be less effective. Additionally, the effects of deceptive URLs and also the limited authentication problems would be reduced. This solution needs further analysis, however, the idea behind it is to remove the decision (passive/active interruption) from the user by asking him a piece of information (interactive interruption) that allows the system to make the decision. In this case, the security will be integrated in the main protocol's flow. Finally we will be converting a complex activity into a task that a ordinary user can perform.

5.4 Consider that the expected behaviour might change under different circumstances

Human behaviour is likely to change under different circumstances. The ability to influence the users' decision making, discussed in Section 3, includes several factors that might influence users' behaviour. Factors such as social conditioning, user's principles, time constraints and shared risk are efficiently exploited by attackers. It is necessary to avoid situations where a user interaction might be made under influenceable conditions. In general, protocol designers should:

- check whether external and internal changes might influence user decisions.
- avoid asking users for decisions when they might be under influence.

In the web browser implementation of SSL/TLS protocol, for example, the dialog (or screen) that asks the users whether they want to accept a certificate or not is implemented in a way that external factors can easily influence users' final decision. If they are under time pressure, for example, this dialog will probably be dismissed with less reasoning. To avoid this situation, warnings should require further checks, such as the domain name confirmation presented in the Section 5.3. By implementing those changes, the user would be forced to "authenticate" the URL, and consequently, avoiding some attacks. An attack exploiting time pressure, in this case, would be less effective.

5.5 Design should prevent user from performing an inappropriate interaction

The system should prevent the user from performing an inappropriate interaction. Norman [20] introduced the concept of a "forcing function", which aims to prevent a user from behaving in any other way than the correct way. Basically, the forcing function prevents users in progressing in their task until they perform an action which must be taken to avoid a failure. Additionally, the forcing function will only enabled the "safe" options when an action is being performed.

Forcing functions prevent errors where a user skips an important step and condition users to progress with the correct (safe) behaviour. To be effective, efforts (cognitive and physical) required to follow the forcing function must be less than the effort required to circumvent it [15]. Thus, protocol designers should:

- attempt to provide only safe options to users, and avoid giving unnecessary (and unsafe) options when not needed.
- avoid drastic changes in the usability due to use of forcing functions. If the impacts are too high, users will try to find ways to avoid the "safe paths".

Following the previous examples, in the web browsers' implementation of SSL/TLS protocol, the way that the decision of accepting a certificate is implemented does not protect users from making an inappropriate decision. In the case of a spoofed website presenting a certificate, the invalid option (accepting the fake certificate) is still available. On the other hand, predicting users' intentions is, for obvious reasons, infeasible. However, it is possible to "ask" users for their intentions and then check if the actions match the intentions (Brustolini and Salomon implemented such mechanism in [4]) before proceeding. The domain name confirmation presented in the Section 5.3 is an example of a forcing function in this case. The user would only be able to proceed if the certificate presented by the server matches with the server the user wants to have access.

As we can see, this function is very useful. However, we must take care with the usability impacts and trade-offs, otherwise users will attempt find ways of dismissing this feature, whenever possible.

6 Conclusions and future work

We have shown that there are many factors that should be taken into account when considering human-protocol interaction. At the same time, there is a wealth of research involving human behaviour analysis and detecting human characteristics that might be exploited by attackers in specific contexts, such as phishing scams and authentication systems. However, despite the existence of similar findings, there is a lack of harmonization regarding the definitions of human characteristics and weaknesses. In this paper, we proposed a set of human-computer interaction overlooked components weaknesses that merges different research findings into a well defined set.

From the first set, we built a set of recommendations to assist designers in the complex task of minimizing security threats from user interaction. The recommendations are based on our findings, related work, empirical analysis and extrapolation from the set of weaknesses presented earlier.

Despite the fact that most of the examples used are based on experiments developed in the key pieces of research carried out to date, further validation of the human-protocol interaction weaknesses and design recommendations against real world systems is an important next step. This will allow us to verify and improve the findings of this work and also the associations among the two sets (weaknesses and recommendations) we propose.

References

1. A. Adams and M. A. Sasse. Users are not the enemy. *Communications of the ACM*, 42:40–46, Dec. 1999.
2. R. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley Publishing, 2 edition, 2008.
3. S. M. Bellovin and M. Merritt. Limitations of the kerberos authentication system. *ACM SIGCOMM Computer Communication Review*, 20:119–132, Oct. 1990.
4. J. C. Brustoloni and R. Villamarín-Salomón. Improving security decisions with polymorphic and audited dialogs. In *Proceedings of the 3rd symposium on Usable privacy and security*, SOUPS '07, pages 76–85, New York, NY, USA, 2007. ACM.
5. L. F. Cranor. A framework for reasoning about the human in the loop. In *Proceedings of the 1st Conference on Usability, Psychology, and Security*, pages 1–15, Berkeley, CA, USA, 2008. USENIX Association.
6. R. Dhamija and J. D. Tygar. The battle against phishing: Dynamic security skins. In *Proceedings of the 2005 Symposium on Usable Privacy and Security*, SOUPS '05, pages 77–88, New York, NY, USA, 2005. ACM.
7. R. Dhamija and J. D. Tygar. Phish and hips: Human interactive proofs to detect phishing attacks. In H. Baird and D. Lopresti, editors, *Human Interactive Proofs*, volume 3517 of *Lecture Notes in Computer Science*, pages 69–83. Springer Berlin / Heidelberg, 2005.
8. R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 581–590, New York, NY, USA, 2006. ACM.
9. J. S. Downs, M. B. Holbrook, and L. F. Cranor. Decision strategies and susceptibility to phishing. In *Proceedings of the second symposium on Usable privacy and security*, SOUPS '06, pages 79–90, New York, NY, USA, 2006. ACM.
10. S. Egelman, L. F. Cranor, and J. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1065–1074, New York, NY, USA, 2008. ACM.
11. C. Ellison. Ceremony Design and Analysis. Cryptology ePrint Archive, Report 2007/399, Oct. 2007.
12. M. L. Finucane, A. Alhakami, P. Slovic, and S. M. Johnson. The affect heuristic in judgments of risks and benefits. *Journal of Behavioral Decision Making*, 13(1):1–17, 2000.
13. P. G. Inglesant and M. A. Sasse. The true cost of unusable password policies: password use in the wild. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 383–392, New York, NY, USA, 2010. ACM.
14. M. Jakobsson. The human factor in phishing. In *In Privacy & Security of Consumer Information '07*, 2007.
15. C. Karlof, J. Tygar, and D. Wagner. Conditioned-safe ceremonies and a user study of an application to web authentication. In *Sixteenth Annual Network and Distributed Systems Security Symposium*, NDSS 2009, Feb. 2009.
16. J. E. Martina and M. C. Carlos. Why should we analyse security ceremonies? First CryptoForma workshop, May 2010.
17. K. D. Mitnick and W. L. Simon. *The Art of Deception: Controlling the Human Element of Security*. John Wiley & Sons, Inc., New York, NY, USA, 2003.

18. C. Neuman, T. Yu, S. Hartman, and K. Raeburn. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 4120 (Standards Track), July 2005.
19. D. A. Norman. Design rules based on analyses of human error. *Commun. ACM*, 26:254–258, Apr. 1983.
20. D. A. Norman. *The design of everyday things*. Basic Books, New York, NY, USA, Aug. 2002.
21. R. Oppliger and S. Gajek. Effective protection against phishing and web spoofing. In J. Dittmann, S. Katzenbeisser, and A. Uhl, editors, *Communications and Multimedia Security*, volume 3677 of *Lecture Notes in Computer Science*, pages 32–41. Springer Berlin / Heidelberg, 2005.
22. J. Reason. Understanding adverse events: human factors. *Quality in Health Care*, 4(2):80–89, 1995.
23. R. Ruksenas, P. Curzon, and A. Blandford. Modelling and analysing cognitive causes of security breaches. *Innovations in Systems and Software Engineering*, 4:143–160, 2008.
24. M. A. Sasse, S. Brostoff, and D. Weirich. Transforming the 'weakest link' - a human/computer interaction approach to usable and effective security. *BT Technology Journal*, 19:122–131, July 2001.
25. S. E. Schechter, R. Dhamija, A. Ozment, and I. Fischer. Emperor's new security indicators: An evaluation of website authentication and the effect of role playing on usability studies. In *In Proceedings of the 2007 IEEE Symposium on Security and Privacy*, SP '07, pages 51–65. IEEE, May 2007.
26. F. Stajano and P. Wilson. Understanding scam victims: seven principles for systems security. Technical Report 754, Cambridge, Aug. 2009.
27. F. Stajano and P. Wilson. Understanding scam victims: seven principles for systems security. *Communications of the ACM*, 54(3):70–75, Mar. 2011.
28. R. West. The psychology of security. *Communications of the ACM*, 51:34–40, Apr. 2008.
29. M. Wu, R. C. Miller, and S. L. Garfinkel. Do security toolbars actually prevent phishing attacks? In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 601–610, New York, NY, USA, 2006. ACM.
30. K.-P. Yee. Aligning security and usability. *IEEE Security and Privacy*, 2:48–55, Sept. 2004.