Maybe Poor Johnny Really Cannot Encrypt – The Case for a Complexity Theory for Usable Security

Zinaida Benenson[*]
Computer Science Department
Friedrich-Alexander-Universität
Erlangen-Nürnberg
zinaida.benenson@fau.de

Gabriele Lenzini
Interdisciplinary Centre for
Security Reliability and Trust
University of Luxembourg
gabriele.lenzini@uni.lu

Daniela Oliveira
Electrical and Computer
Engineering Department
University of Florida
daniela@ece.ufl.edu

Simon Parkin
Department of Computer Science
University College London
s.parkin@ucl.ac.uk

Sven Uebelacker
Security in Distributed
Applications
Hamburg University of Technology
uebelacker@tuhh.de

"I continuously go further and further learning about my own limitations, my body limitation, psychological limitations. It's a way of life for me." (Ayrton Senna).

ABSTRACT

Psychology and neuroscience literature shows the existance of upper bounds on the human capacity for executing cognitive tasks and for information processing. These bounds are where, demonstrably, people start experiencing cognitive strain and consequently committing errors in the tasks execution. We argue that the usable security discipline should scientifically understand such bounds in order to have realistic expectations about what people can or cannot attain when coping with security tasks. This may shed light on whether Johnny will be ever be able to encrypt. We propose a conceptual framework for evaluation of human capacities in security that also assigns systems to complexity categories according to their security and usability. From what we have initiated in this paper, we ultimately aim at providing designers of security mechanisms and policies with the ability to say: "*This feature of the security mechanism X or this security policy element Y is inappropriate, because this evidence shows that it is beyond the capacity of its target community*".

CCS Concepts

•**Security and privacy → Usability in security and privacy;**

Keywords

Usable security models, human capacities.

1. INTRODUCTION

Usable security as a discipline is almost twenty years old – or older if one considers the principles A. Kerckhoffs wrote in 1883 in *La cryptographie militaire* [42] as the first usable security statement – and still Johnny cannot encrypt.

In the modern literature the earliest conceptual paper on usable security is that of Zurko and Simon, published in NSPW 1996 [92]. It discusses the necessity of *psychological acceptability*[1], summarises the scarce usable security research at that time, and points out design directions for user-centred security mechanisms. Since then, many works have investigated usable security. In 2005, Cranor and Garfinkel [23] edited the "Security and Usability", an omnibus volume collecting work done in the field up to that time. In the same year Zurko [91] discussed the many open problems (and some solutions) in usable security and formulated the grand challenge of the field: *"Give all users (including developers, administrators, and end-users) security controls that protect them, their systems, and their privacy, that they can use appropriately in the dynamic, pervasive computing environments of the present and the future."*

Almost 10 years have passed since this challenge was stated, and still successes in usable security are not being seen on a large scale. As Herley puts it [36]: *"While usable security has had many successes in pointing out the failings of security UI, progress has been slower at providing actionable alternatives."*

Considering the works published in the last year (2014), we can see that Johnny still cannot encrypt [69], choose strong passwords [44, 79], comply with browser warnings [28], identify suspicious links in messages [8, 17], learn from security training [17], or write secure software [57]. That Johnny is so "inept" with security technology frustrates the security community. One wonders whether the community is lacking the required inventive spark to develop solutions that can help Johnny, or whether there is something intrinsically difficult that prevents Johnny from performing security tasks as expected by those who design them, be they usable security researchers or security practitioners.

Perhaps it is this intrinsic difficulty that leads researchers and Johnnys alike into a spiral of frustration and blame: Researchers,

[*]Authors listed in alphabetic order of last names.

NSPW '15, September 08-11, 2015, Twente, Netherlands

© 2015 Copyright held by the owner/author(s). Publication rights licensed to ACM. ISBN 978-1-4503-3754-0/15/09…$15.00

DOI: http://dx.doi.org/10.1145/2841113.2841120

[1]Psychological acceptability is a term coined in 1975 in Saltzer and Schroeder's classic paper on design principles for secure systems [72]: *"It is essential that the human interface be designed for ease of use, so that users routinely and automatically apply the protection mechanisms correctly. Also, to the extent that the user's mental image of his protection goals matches the mechanisms he must use, mistakes will be minimised. If he must translate his image of his protection needs into a radically different specification language, he will make errors."*

unable to influence Johnny's secure behaviour, label him "the weakest link of the security chain"; Johnny, annoyed by the excessive security demands in his way, calls them "security paranoids", since they don't realise that as much as he values security, he needs to send that email whether it is encrypted or not. To develop a healthier approach to the problem, it seems time for researchers to ask the question: *Is usable security possible for all tasks?*

1.1 Respecting Human Capacities

Development of security mechanisms and policies for the general public and employees in corporations alike currently happens in a way that relies on "best practices" rather than on scientific evidence [80]. This can result in mechanisms and policies that are *beyond capacity* of people, that is, beyond their ability, experience, or understanding. Such mechanisms disregard fundamental principles of how people function, what they might reasonably refuse to do, and what they are physically and mentally able to achieve.

The technical security community historically thinks that it is the responsibility of users to operate at the capacity required by a security mechanism. If the corresponding security requirements are not achieved, it is seen as the fault of the users.

The usable security community, in contrast, argues that security requirements can be met through careful design of usable security solutions. For example, in a recent paper [62] leveraging behavioural science to mitigate security risks, Pfleeger and Caputo argue that *"if humans using computer systems are given the tools and information they need, taught the meaning of responsible use, and then trusted to behave appropriately with respect to cyber security, desired outcomes may be obtained without security's being perceived as onerous or burdensome"*. If the corresponding security requirements are not achieved, the usable security community tends to attribute this to the failure of the security designers.

The position of both communities needs to converge. The technical security community has to acknowledge what the usable security research tells us about human interaction with security technology; the usable security community has to review its approach, because providing tools and information help little if we overlook whether people is being asked to execute tasks beyond their capacity. For example, it seems that, on average, no amount of advice on the personal use of passwords will enable people to maintain unique random passwords for a typically large number of accounts [10,30]: although password management tools could help, such usable tools are rare and user acceptance of these tools is low [54].

Although evidence of other infeasible tasks is yet to be developed, we think that various unusable security solutions hint at their existence. Consequently, we propose to investigate which security tasks are within or beyond the capacity of their *intended* users. For example, trained users can encrypt their emails while the general Internet population cannot – this may be because understanding Public Key Infrastructure (PKI) well enough to be able to exchange encrypted emails would require a critically large part of the Internet population to become fully-fledged security experts.

There are at least two advantages in determining which tasks are beyond capacity of users. First, one could substitute an infeasible task with a task that ensures the same level of security but is within capacity. Second, one could help users to perform the tasks, for instance by supporting them with mnemonic techniques, with complementary security technologies, or with training. We note, however, that also the feasibility of these additional measures have to be examined. Whichever way one chooses, if users are alleviated from executing security tasks beyond their capacities, they will perceive less burden and be more willing to act securely [4].

We give two examples of such benefits. If we could gather reliable scientific evidence that, for example, a typical home computer user is not able to understand PKI-based systems to the extent required for execution of the corresponding tasks, the users will not be blamed for not encrypting their emails or for not complying with SSL certificate warnings (if this has been determined to be an appropriate task for end-users to enact). In this case, with knowledge of the human capacity to approach the task, the security community will have additional motivation and evidence to redesign the corresponding systems to achieve more consistent outcomes. In the case of spear phishing emails, if there was an upper bound to an individual's ability to detect deceptive elements in emails, then a good strategy would be to develop and adopt other security mechanisms.

1.2 Measuring Complexity of Usable Security

To tackle the problem of user capacities in a way that is fair to both users and designers, we need to approach the question in a structured manner. Our goal is to understand which tasks are so difficult for the users as to be considered beyond their capacity. This effort will help us identify upper bounds on the level of usability achievable by a system containing such tasks.

This way of reasoning is similar to the evaluation of time and space complexity of algorithms in the complexity theory [21]. By measuring an algorithm's complexity and by observing that exponential-time procedures are inefficient, one can objectively predict the algorithm's performance and the size of problems one can effectively solve using it. We do not aim at setting up a mathematical complexity framework in usable security; we do not think that people can be treated in the same way as Turing machines. However, having an estimation of task complexity in terms of human resources will help establish evidence-based expectations on the effective usability of a system.

There are three points to discuss if we intend to hold the analogy between algorithm complexity and system usability. We need to establish (1) a model of computation, (2) a measure of computational complexity, and (3) a notion of intractability [31,59].

1. The model of computation in usable security is the *intended user* as a representative of a group of users with specific cognitive and physical abilities. Here, we also consider that humans can augment their natural capacities. For instance, they can use a memory device such as a mental story-board to memorise more things, a USB-token to hold a password, or they can rely on *transactive memory* [86] when working in groups. In such cases, our model of computation is the human and the device or the group of humans, since these are the entities that execute tasks.
2. A measure of computational complexity is the amount of internal resources a human spends to execute a security task, that is, the required physical and cognitive effort.
3. When the demands of a task exceed human capacities, people experience strain, which can lead to mistakes or abandonment of the task. This could be a definition of intractability, meaning that a task, because of its cognitive or physical demands, is "beyond human capacity". Note that we may be unable to assess a task's complexity if we have not enough evidence to draw reliable conclusions.

We are interested in clarifying the implications of "usability complexity" for the overall *usable security* of a system. In Section 6 we propose and discuss a conceptual framework that distinguishes different degrees of usable security in terms of two variables, "usability" and "security". The framework relies on (a) a measure of a system's usability in terms of "tasks within" and "tasks beyond" user capacity and (b) an assessment of a system's security in terms of satisfied requirements. We are aware that these measures of usability and security are very coarse-grained and need to be refined in the future; hoever, even so, they suffice to provide a clear preliminary classification of systems in terms of their usable security.

We can now formulate our research question as follows: *In usable security, can we determine whether security tasks are within capacity, and which security tasks are beyond capacity?*

Before considering that what we can do if we knew that tasks are within or beyond capacity, we should provide evidence that human capacities for security-related tasks can be measured after all. We intend to call on outputs from other research areas and to systematically apply them to the security domain. For example, neuroscience research [75] has documented for decades the features of human memory in forgetting (omission) and distorting (commission) facts. Psychologists have known for decades that humans behave like communication channels [55], as much as the channels in Shannon's information theory [78], and that they can, on average, transmit reliably and without "getting confused", only 2.5 bits, which corresponds to approximately 7 chunks of information. All these results are based on experimental evidence, meaning that human capacities can be measured in these disciplines.

Contribution.

This paper makes the following contributions:

1. It introduces a new paradigm of considering usable security from the task complexity point of view, acknowledging the existence of intractable and hard tasks in terms of human cognitive and physical capacities.
2. It proposes an ECG (electrocardiogram)-based methodology for measuring human capacities, which can be refined and used to catalogue capacities for common security tasks.
3. It proposes a framework with definitions of complexity categories for usable security. The framework is expected to be refined in future work with the help of the community of researchers and practitioners.

This paper is organised as follows. Section 2 discusses the notion of usable security. Section 3 presents examples of human capacities from a psychology and neuroscience viewpoint. In Section 4 we provide examples of complexities for security tasks, specifically password authentication, USB security, and email encryption. Section 5 discusses an ECG-based methodology for measuring human capacities for security tasks. Section 6 presents an initial framework for complexity of usable security. Section 7 discusses our ideas in the light of related work. In Section 8 we present our concluding remarks and future work.

2. USABLE SECURITY

In this section, we clarify our working definition of "usable security", as there seems to be no unified definition in the community. For example, a recent book by Garfinkel and Lipford [32], summarising research in usable security from the very beginning to 2014, provides definitions for security and for usability, but does not define the term "usable security" *per se*.

Meanwhile, as widely acknowledged in the community, usable security is not just the union of the definitions of its separate parts, security and usability [46,48,74,81,85]. It requires a careful adaptation of the usability concept to the security domain, as discussed by Whitten and Tygar in their classical work on usable email encryption [87]: *"Usability necessarily has different meanings in different contexts. [...] In a security context, our priorities must be whatever is needed in order for the security to be used effectively"*.

In Section 2.1 we define usability and consider the implications of this definition for the security domain; then, in Section 2.2, we discuss the role of usability in achieving security. These considerations are later reflected in the usability measurement framework presented in Section 5 and in the complexity framework for usable security in Section 6.

2.1 Defining Usability of Security

Although many classical definitions of usability exist, for the purpose of this work we consider the ISO 9241-11 definition [38]: *Extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.*

The advantage of the ISO definition is its operationality, as it tells us how to test the usability of a product: we first should specify users, their goals, tasks and context of use, and then test the task execution for effectiveness, efficiency and satisfaction. We note that although satisfaction can only be tested with the *intended* users, effectiveness and efficiency can be also tested without involving the users, *e.g.,* using cognitive walkthroughs, heuristic usability evaluation or GOMS modelling [56,88].

With a definition of usability, we next explore questions that arise when applying this definition to security systematising similar considerations previously made by various authors [46,48,74,81,85].

Notable stumbling blocks in the ISO definition are the notions of "product" and "goals". Security measures are rarely a stand-alone product, rather they are integrated into a system with the goal of protecting the system and its users from attacks. For example, classical protection goals are confidentiality, integrity and availability of data and other resources. Moreover, security goals are usually not primary, but secondary goals of the user.

For example, the primary goal of sending an encrypted email is communication, and the secondary goal is confidentiality. The product is an email client with an integrated email encryption capability. Can (and should) the usability of the email client be evaluated separately from the usability of the email encryption functionality? Is the user always able to determine whether the goal of encryption is achieved? When encryption software fails, this failure is often not visible [32, p. 55].

The authentication service of a bank's online portal is a very important protection measure for its customers, and should be effective and efficient. However, what about satisfaction? Should customers be satisfied with a two-factor authentication mechanism that delays their primary task of making a money transfer, however easy the mechanism might be to use? In this sense, rather than satisfaction, minimal dissatisfaction is a more realistic positive outcome.

The question of handling errors is also not straightforward. If confidentiality should be maintained, an error that exposes the secret to the attacker cannot be "corrected" anymore (Whitten and Tygar [87] call this the "barn door property").

Moreover, as the security protection tasks often do not serve the primary goals of the users, usability evaluations should consider users in the accomplishment of their primary tasks in the context that is natural for the system usage.

Finally, context of use in the security domain also includes the attacker. Garfinkel and Lipford [32] formulated attacker modelling in usable security as one of the most important research challenges in the usability evaluation of security and privacy measures. The many questions that arise are *e.g.,* how to account for attacks in evaluation methods, how to choose which attacks to include, how to approach the ethical side of user tests involving attacks and deception, and how to manage the validity of the results of user tests (*e.g.,* whether users be more or less vigilant in a test environment).

Thus, although the usability of security may be defined using the same words as in the above ISO 9241-11 definition, some of these words might have different or additional meaning:

Usability of a security mechanism or policy is the extent to which this mechanism or policy can be used by specified users to achieve the specified security goals (some of these goals being invisible or secondary goals to the users) and the specified primary goals with effectiveness, efficiency and satisfaction (or at least with low dissatisfaction) during the execution of the specified primary user tasks in a specified context of use (including the specified attacks).

We use this definition in Section 5 when presenting a usability measurement framework, and in Section 6 when presenting a complexity framework for usable security.

2.2 Relationship of Usability and Security

In 2004, an influential article by Avizienis et al. [2] unified the basic concepts of dependability and security with the goal to facilitate communication and cooperation between the corresponding research communities. The authors position *dependability* and *security* as fundamental non-functional system properties alongside the functional properties of *functionality* and *performance*.

In particular, dependability is described as a concept integrating *safety* as an important attribute, where safety is defined as *the absence of catastrophic consequences on the user(s) and the environment* [2, p. 13]. Security is defined as *the absence of unauthorised access to, or handling of, system state* [2, p. 23].

The similarities between security and safety have been examined by the usable security community [15]. In particular, different methods of usability analysis constitute important techniques for assuring safety of critical systems. Revisiting the role of terminology, we note that the term "usable safety" does not exist, and other terms, *e.g.,* "human factors" or "human error", are used instead in the field of safety [19, 67, 88]. It seems that usability can be considered as an integral part of the safety discipline, where both technical and human factors approaches are used to prevent system failures. The complexity framework for usable security presented in Section 6 considers usability and security as connected system properties in a similar manner.

3. HUMAN COGNITIVE CAPACITIES

Psychology and neuroscience researchers have studied human cognitive capacities extensively, and showed that cognitive strain may cause people's committing errors. In security, this can introduce vulnerabilities into the systems and processes people interact with. In this section we review psychology and neuroscience literature on human cognitive capacities, and discuss implications of this research for cyber security.

In the following, we consider the information-processing model of cognition. Alternative models for cognition also exist, such as situated cognition [16], activity theory [47], and distributed cognition [12]. These models view human cognition integrated with the external world. For example, Hollan *et al.* [12] present the distributed cognition paradigm where cognitive processes and the external world interactions are interconnected and considered as a whole. For the present, we leave these alternative models out of scope, although in the future they can also be considered within the framework that we develop in this paper.

3.1 Capacities of Memory

Schacter [75, 76] identified seven memory shortcomings, called "sins" in his papers: transience, absent-mindness, blocking, misattribution, suggestibility, bias, and persistence.

The first three sins involve various forms of forgetting. *Transience* involves the loss of memories with time. Memories that are not retrieved and rehearsed dissipate over time and the loss can occur, depending on the level of use, on a scale of seconds, hours or days. *Absent-mindedness* entails lack of sufficient attention at the time a memory is being encoded or retrieved. The classic example is when people forget where they put their car keys or glasses. *Blocking* occurs when some information has been encoded so deeply that it may sometimes be temporarily inaccessible, even if people are provided with cues related to the item. Blocking is especially pronounced in older adults.

The other four sins are "sins of commission", *i.e.,* some form of memory is present, but is attributed to an incorrect person, time, or place. *Misattribution* involves correctly remembering a fact from a past experience but misattributing it to an incorrect source. *Suggestibility* refers to illusory memories occurring in response to suggestions that are made when one is attempting to recall an experience that may or may not have occurred. In other words, it refers to the tendency of incorporating information provided by others. *Biases* are a person's pre-existing knowledge and beliefs that influence memory encoding. Finally, *persistence* involves remembering a fact that one would prefer to forget.

The way these memory features interact depends on a variety of factors, including how much attention the person is giving, how novel and interesting the experience is, and the kinds of emotions that are evoked by the information or stimuli. Schacter and many neuroscientists [77] argue that these features are actually the cost of an adaptive system that has been hardwired to get the gist of a situation and not be cluttered with unimportant details.

Omission and commission sins correspond well to the error classification into "omission errors" and "commission errors" [67] and have important implications for computer security, for example:

1. Transience: forgetting passwords, steps of authentication routines, meaning of warnings, key points of awareness training;
2. Absent-mindedness: clicking on a link in an email without paying attention; forgetting laptops or smartphones in public places; forgetting an authentication token at home or in some other place (and not being able to recollect where it is);
3. Blocking: not being able to recollect passwords or PINs that are usually well remembered, but just slipped the memory in a concrete situation, for example under stress or observation;
4. Misattribution: falsely recollecting seeing a person on the premises of the company although actually the person was seen in some other context (e.g., in a shop or on an exhibition), and thus letting them to tailgate; email scam that includes the (necessarily incorrect) customer ID of the victim in an attempt to make the victim "recognise" their ID, believing in the information implanted by social engineers in spear phishing e-mails;
5. Suggestibility: agreeing to the course of events as suggested by a social engineer ("we met last week in this meeting, where X said this and Y did this, do you remember?") and thus complying with his/her request for information or action; email scams that mention past communication in the email subject, such as "Re: request No. 23019";
6. Bias: falsely remembering after an awareness training that a fraudulent email can only come from an "unknown" person and thus not being suspicious of an email with a spoofed known sender;
7. Persistence: remembering an old password or PIN instead of a new one; remembering that a former colleague is still working for the company (although the knowledge that it is not so is actually present) and talking to him/her about confidential things.

3.2 Human Information Processing Capacity

In the classic 1956 psychology paper George Miller [55] discussed the limits of human capacity to process information, relating this theory to Shannon's information theory [78].

Miller compared the human with a communication channel and stated that when the amount of input information is increased, transmitted information will increase first, and will eventually level off at some *asymptotic value*. Miller called this asymptotic value a person's *channel capacity*, or the greatest amount of information that the person can process from a stimulus. It is the *upper bound* on the extent to which the person can match their responses to given stimuli.

In cognitive psychology, cognitive load is the total amount of mental effort used by the human working memory. Psychology literature documents classic examples of tasks that were empirically found to cause cognitive strain in people. In other words, these tasks cause people to operate at a level that is above their channel capacity. For example, Pollack [65] conducted an experiment in which participants were asked to identify tones by assigning numbers to them. Participants did not have difficulties in identifying up to four tones. With five or more tones, mistakes were common. Pollack also conducted experiments that involved identifying loudnesses, saline concentration, and points in a line and in a square.

Miller [55] analyses data from Pollack's experiments in terms of the number of bits processed by the participants, considering that one bit of information provides the possibility to distinguish between two possible alternatives. On average, the transmitted information increases linearly to about 2 bits and then reaches the upper limit of 2.6 bits with standard deviation 0.6, which is about 7 alternatives ± 2. Miller hypothesised the existence of a limitation on our nervous system that keeps our capacities in that range.

The processing capacity varies systematically according to the individual differences in people. For example, there is broad evidence of age-related decline in a wide spectrum of cognitive abilities [53], which causes the channel capacity of older adults to be reduced. Also, experts on a task have more channel capacity for that task compared to novices [89, p. 208ff].

We think that also for every security task there will be an asymptote for human capacity that will put a limit on what can be expected from people. Well-designed usable security solutions should allow humans to execute security tasks while minimising the draw on their capacity.

4. HUMAN CAPACITIES IN SECURITY

In this section we illustrate the role of human capacities in password authentication [73, 82], USB device decryption [3, 61], and email encryption [33, 87], based on related literature and accounts of technology use by employees in organisations.

The security routines described in these examples are arranged as sequences of tasks, called *traces*. Users may be impacted during trace progresses and cognitive or physical load builds up, even if the single tasks are completed successfully.

If the cumulative demand is perceived to be too high, users may try to find an alternative to the security task. Critically, if users perceive a task to be too complex or unworkable, they will try to modify that complexity themselves, and create workarounds [4]. Individuals may utilise the security knowledge and resources they have available to them to create what they believe to be a more usable security control, without intervention from system designers [43]. Where a task is sidestepped or altered, weaknesses in security may appear.

As a consequence of *transience* as described in Section 3, a workaround may be sought if a user forgets how single tasks combine into a sequence (if no care has been taken to provide a sequence that is easy for non-experts to remember). If a user decides that a security task demands the seemingly impossible (e.g., recall a security question configured years prior – this may be an example of *blocking*), this security task will be beyond the user's personal capacity. These are arguably tasks which do not consider the realities of the human mind, and are often attributed to a failing of users rather than to an oversight by security designers.

4.1 Example 1: Password Authentication

An example of a password authentication routine is shown in Figure 1. Note that this is only the security routine of entering a password and being informed of the outcome (successful or unsuccessful authentication). In reference to Figure 1, individual tasks are identified based on their capacity to impact – and potentially overload – the user, and for responses to those tasks to modify the overall trace of security tasks that make up the security routine.

Security Prompt: Task-switching, in itself, may involve *persistence* (Section 3), where elements of a recent task become confused with the current task. A user may authenticate in advance so that they can focus on a primary task unimpeded, or otherwise "batch" their primary tasks together to reduce task-switching. A choice may be made to forgot the security task altogether if the (perceived or actual) effort it demands is regarded as excessive (exceeding the cognitive load). The individuals may feel embarrassed or become stressed if they are subject to *transience*, and cannot recall how to enact the security routine. This can further impact adjoining tasks. A user may give up on the task entirely or use a workaround, where the former results in missed opportunities [4] and the latter potentially weakens the security qualities.

Recall Password: The number of accounts a user has to manage can make it difficult to recall which username and password applies in a given context. *Misattribution* (Section 3) such as this can be problematic if the user has employed a coping strategy of using the similar passwords across multiple accounts to manage what they perceive as an excessive number of accounts [30]. A password may be partially remembered, which is still not enough to afford access. If the user is working within an organisation, they may have been forced by policy – or automated security systems – to use a password that they believe is too complex to remember (especially amongst a set of passwords of potentially equivalent complexity). Even where a person is able to recall a string of a given complexity, security system designers/managers may not consider that this string is one amongst many that they will have to manage [30]. A person's ability to create and remember a series of passwords, and recall the right password for the right system, is in this case their *channel capacity*, where that capacity is often assumed by system designers to be unbounded.

For employees in an organisation, there may be dedicated self-service or staffed helpdesk support for resetting passwords. If a password is forgotten, a reset request may be addressed quickly or may take a long time, losing the engagement of the user and promoting *absent-mindedness*. An individual may attempt to reduce their authentication effort by employing a password manager (e.g., in browser context) or a password "caching" agent such as `ssh-agent` or `gpg-agent`. A password manager may be sanctioned by the organisation, or introduced by the user as a personal coping strategy [43], in either case acting as a potential security weak point. Alternatively, a user may choose to duplicate a password across multiple accounts as a means to personally manage their

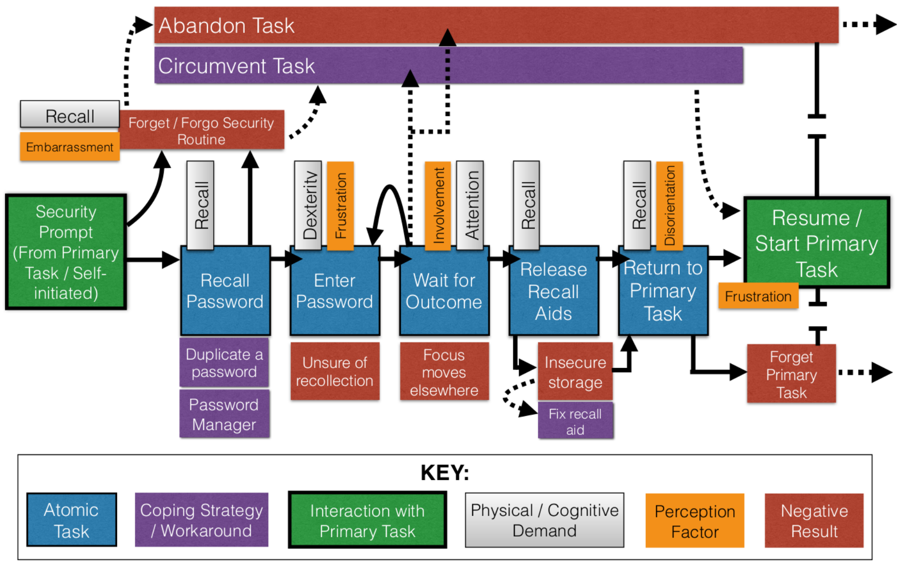

Figure 1: Password authentication process.

channel capacity. As a workaround a password may already be stored somewhere local to the machine (e.g., on a sticky note under the keyboard), or even on the same machine (e.g., in an electronic file), limiting the need for recall.

Enter Password: People may feel embarrassed if they have to make multiple entry attempts, especially if their physical dexterity is called into question – the impact ageing (for instance) has upon the process is then both mental and physical. Password entry on a touch-screen device may take longer than on a keyboard, depending upon the feedback that the device produces during typing. Password entry may potentially happen in parallel with password recall, character by character, where longer passwords can demand more of the user. To aid recall of a single password amongst many, a user may recall the password creation rules governing accounts for a particular system amongst many, inviting *misattribution* and challenging their *channel capacity*.

Wait for Outcome: A period of inactivity is an activity in itself, where the user is expected to hold their attention on the task. There is potential for *absent-mindedness* if the user is not actively involved in the progression of the task. There may be an allowed number of authentication attempts, where reaching that limit forces a break from the security routine. If difficulties appear – or are expected – an individual may abandon or circumvent the task, employing an ad-hoc workaround or repeatable coping strategy.

Release Recall Aids (optional): A recall aid such as a sticky note or written password may be left near the user's area of work or kept within easy reach. The act of creating, updating, or securely storing a recall aid can further delay completion of the security task.

Return to Primary Task: If periods of inactivity require the users to re-authenticate, they may be returned to the task "Recall Password" at any time before or after returning to the primary task, confusing the *persistence* of different tasks in memory. Once authenticated, an individual may struggle to recall what they wanted to do. Moving between a primary task and a security task may not simply be a case of switching between the two, but of completing a *blocking* security task and then recalling where they were in the previous task, which can demand effort and leave the individual disorientated for a time. Any stress felt during the security routine may linger as frustration, carried into any tasks that follow.

Abandon / Circumvent Task: A habit may be reinforced where the individual refuses to engage with the task again or decides to engage in a workaround behaviour or coping strategy as a preference, instead of the advocated behaviour.

4.2 Example 2: USB Device Decryption

The usage of an encrypted USB storage device, where the data on the device have to be accessed on various computers across a range of locations [3], may also challenge users' capacities.

We assume that encryption software must run on a computer when an encrypted USB storage device is connected, requiring a password. As in the authentication example, there will be a **Security Prompt** for entering a password, here to activate the decryption process. The user will need to **Recall Password** and **Enter Password** (and subsequently **Release Recall Aids** before a **Return to Primary Task**). However, given the mobile nature of USB storage devices, the location in which these tasks occur may be potentially insecure (e.g., in transit or at a business event where shoulder surfing for the decryption password may be feasible). *Channel capacity* of the users is especially critical here, as they would have to remain mindful of their immediate surroundings while also working with the device, constantly evaluating changes in their environment in order to avoid shoulder-surfing or tampering with the device. The time cost of decrypting the device can promote *absent-mindedness*.

In order to prevent damage if the USB stick were to be lost or stolen (*absent-mindedness*), users would be strongly encouraged to enforce encryption of stored data when outside of company premises, challenging their *cognitive capacity to understand* if the device is genuinely encrypted, especially if the organisation is not clear about what constitutes appropriate encryption.

The device and decryption software may not work correctly with the computer it is connected to: the computer may be unsupported by the employer organisation, or the employee may recall the decryption procedure incorrectly (*transience* or *bias*). The user may then find some ad-hoc means to access the device, requiring security and technology expertise alongside their limited security training, while the machine itself may in fact not allow them to access the device in any other way, promoting frustration.

Equally, if the individual has been instructed to contact a support team in such instances, they would have to recall the exact process for reaching the support team (with potential *misattribution*) – this may all occur while under time pressure to access the device. Prior negative or stressful experiences of interacting with IT support may influence the user to **Abandon / Circumvent Task**.

Friendly individuals who have malicious intentions – or lack of awareness of security implications – may offer seemingly easier means to access content (e.g., encouraging the user to access corporate systems through their own computer, an example of *suggestibility*). Former colleagues may offer to help in the same way, as if they are still co-workers (*persistence*).

Employees may balance the availability of data against the confidentiality risk to their organisation if they are especially fearful – and distressed – by the possibility of not having guaranteed access to content [4]. Thus, a user may **Abandon / Circumvent Task** by choosing not to encrypt the device at all (if encryption is not enforced), or send the files to themselves via unencrypted email.

4.3 Example 3: Email Encryption

We hypothesize that any email encryption system that requires understanding of the public key infrastructure (PKI) will be beyond capacity for a critical mass of non-expert Internet users, as suggested by the analysis presented below.

End-to-end email encryption has been considered the holy grail of usable security since the seminal investigation of PGP usability by Whitten and Tygar [87]. This first study uncovered the inability of untrained users to execute the typical tasks required to set up and use PGP. Whereas the study's authors attribute this failure to the unusable user interface of the PGP email client, Garfinkel and Miller [33] suggest that the users were unable to grasp the key certification model of PGP. Moreover, they also criticise the need for third party certificates for the usage of S/MIME, as it leads to usability and understanding problems, including the necessity to react to certificate warnings for unknown certification authorities.

The "Johnny 2" study [33] uncovered yet another PKI-related problem: in a system that relies on self-signed certificates, an active attacker can easily convince the users to trust the attacker's key (*suggestibility*), which can also be attributed to a fundamental misunderstanding of security guarantees the PKI provides in this case.

Not surprisingly, one of the most recent email encryption systems specifically designed with usability in mind [71] does not use PKI at all and relies on key escrow in order to hide security details from the users.

A qualitative study of attitudes to and understanding of email encryption by Renaud et al. [69] also confirms that the non-usage of encrypted email is not entirely due to the usability issues, but also results from misconceptions and incomplete understanding of email architecture and email security.

5. MEASURING HUMAN CAPACITY FOR SECURITY TASKS

Is the cognitive effort for security tasks measurable? We hypothesise that the answer is 'yes'. Here we propose a proof-of-concept methodology for the corresponding experimental design [49].

This methodology has two goals. The first is to allow security researchers and practitioners to discover the level of cognitive effort required by a security task, especially whether this effort is within human capacity. The second goal is to design a catalogue of capacities for common security tasks that can be consulted by the security community when designing security mechanisms and policies. In the following, we first discuss the measurement of single security tasks (Section 5.1), then generalise to the measurement of sequences of security tasks, or *traces* (Section 5.2), and then discuss incorporating primary task, users' context and attacks into the measurements (Section 5.3).

5.1 Measuring Single Security Tasks

We consider a set T of common security tasks or expected behaviours, such as authenticating using a password or looking out for suspicious emails and websites. The tasks in T may have different levels of detail, depending on the measurement goal and the measurement method. For example, password authentication or USB device decryption can be broken into tasks as described in Section 4. Additional examples of tasks may include checking whether the email address of the sender and the name of the sender match in an email, or looking for the lock icon or "https" when providing sensitive information online.

The cognitive load for tasks that involve user interaction with some interface elements will also depend on the implementation of these elements (e.g., noticing and understanding the lock icon), such that the description of the user interface should be included in the task description. For some other tasks, such as recalling a password, the user interface may or may not not play an important role. The goal is to measure the cognitive strain caused by these tasks. One of the challenges is designing an experiment that isolates the targeted task or behaviour in a meaningful way.

We also consider the intended users U for the tasks in T. The users may differ in their capacities, for example according to their age, gender, education, work experience or other demographic characteristics, therefore the specification of the intended users is important. For security tasks that should be executed by any Internet user, *e.g.,* email encryption, the specification U would include age, gender and education distribution of the Internet population by country of residence, and the measurement results might differ accordingly. For more specific security tasks, such as, *e.g.,* finding buffer overflows in pieces of code, U might specify level of programming experience in years or lines of code.

Consider a set A_i of possible inputs that must be processed by U for each task $T_i \in T$ and for each input $a_j \in A_i$. For example, suppose T_i is the task of recalling a password. Depending on the requirement of the authentication system, people might be asked to recall a password of just four digits, or at least eight characters, or at least eight characters from three of four character classes (while requiring that the password not match any words in a cracking dictionary) [44]. We state the following null and alternative hypothesis for a task $T_i \in T$ and an input $a_j \in A_i$:

- $H0_i$: the execution of T_i does not cause cognitive strain in U for the input a_j.
- $H1_i$: the execution of T_i causes cognitive strain in U for the input a_j.

The process of formulating and testing the hypotheses requires an understanding of what cognitive strain is and how to measure it.

In the psychology and neuroscience literature, there are several works addressing how to measure mental effort and cognitive strain. For example, Paas and Van Merienboer [58] developed a relative condition efficiency to measure perceived mental effort. Granholm et al. [34] propose the use of pupillary responses to measure cognitive strain. Pupillary responses increase with the cognitive load, reach an asymptote at the limit of human capacity and then decrease at cognitive strain. Cooper-Martin [20] used self-reports on time and cognitive strain.

Measurement of cognitive strain is also widely used in cognitive engineering for all stages of human information processing (attention, memory workload, cognition) and applied to such complex tasks as distracted driving (*i.e.,* driving while eating, conversing with the passengers or making a phone call) [89].

We leave an in-depth investigation of different measurement methods out of scope of the present paper. Our goal is to present one possible measurement framework, whereby we do not claim to present the most convenient or the most up-to-date one.

A relatively convenient and affordable example of a measurement method for cognitive strain is ECG (electrocardiogram), which is a test that records the electrical activity in the heart. An ECG device records signals across multiple heart beats and produces a strip that can be interpreted by a healthcare professional or specialised software. There are several types of devices for measuring ECG via wrist or waist band in the market [64], and there are MATLAB-based open source software to interpret the various ECG components [9, 84].

Several studies [37,60] identify a relationship between heart rate variability (HRV) in ECG recordings and stress caused by strain (cognitive or physical). By using a baseline of HRV measurements corresponding to cognitive strain conditions, one can then discover whether any $T_i(a_j)$ is beyond human capacity.

Stress is a feeling of strain and pressure and occurs when people feel they are unable to manage the demands placed on them, which can be cognitive or physical tasks, deadlines, major life events etc. [11]. In these cases the demands of the situation outweigh the body's ability to cope. Regardless of the type of straining task or situation, the psychological feelings or "flight or fight" induced by stress causes the same adverse conditions in the body, be it a physical threat or impossible deadlines [1].

Measurements of security-related tasks must then be calibrated against a baseline of cognitive strain. We propose to select a task (not necessarily security-related) on an input known to cause cognitive strain in people and measure people's ECG in such conditions. One candidate baseline task is Pollack's test of recognition of different pitches [65] described in Section 3, where HRV measurements can be introduced. Baseline measurements may be amassed gradually, dictated by research goals. Collection may focus on specific demographic characteristics, such as age groups for instance, as literature shows that people's cognitive skills decline with age [53].

With the baseline for ECG measurements for cognitive strain, one can design experiments to discover whether someone performing a security task T_i for input a_j experiences cognitive strain. If the HRV recordings for $T_i(a_j)$ indicate cognitive strain, this means that performing $T_i(a_j)$ is beyond this person's capacity.

5.2 Measuring Traces

We now discuss how to measure sequences of tasks, or traces, for the common security mechanisms as described in Section 4. That is, we would like to know how the combination of tasks with known upper bounds on capacities combine into a security mechanism. Is the cognitive strain adding up when the sequence of tasks is executed? Does a specific task in a trace cause cognitive strain? Is it possible to predict upper bound on capacity for a trace if we know upper bounds for each of its tasks?

The measurement procedure of a trace is similar to the measurement of individual tasks. However, careful experimental design is needed to determine the boundary between tasks. For example, how can one determine the time interval in which the user recalls a password? Should the beginning of this time interval be fixed at the point in time when the password prompt appears? There is no guarantee, however, that the user notices the prompt at the exact point in time of its appearance. One can use an eye tracking device to measure this, or one can try to approximate from when the prompt appears (perhaps based on data from previous experiments). In this way, measurement of a trace may incorporate a number of measurement devices.

Password recollection may begin when a person starts typing in a password, but at that moment recollection may not be complete. Recollection and entry are not then a sequence, but a parallel execution of tasks. Moreover, if a person under observation types a letter and then corrects it, the cause may be an accidental slip on the wrong key, or a wrongly remembered character that was corrected "on the fly".

This example shows that the desired goal of measuring the combination of tasks may not always be achievable, or demand substantial resources. The considerations of what can and what should be measured are highly dependent on the measurement goals and of the security mechanism in question.

5.3 Context, Primary Task, and Attacks

As discussed in Section 2.1, usability tests of security mechanisms should include the specification of the primary task and of the adversary, when appropriate. Also the context of use (Is the working environment noisy? Are there many interruptions?) is traditionally considered in usability testing, and especially with safety-critical tasks such as driving, aircraft control, medical equipment operation or nuclear plant operation [67].

Ideally, we would like to know how the human capacities in security tasks change when a person is focused on their primary task, or interrupted, or when the system is under a technical or non-technical attack. Examples of studies in usable security (that did not measure capacities, however) are the study of S/MIME usability by Garfinkel and Miller [33] and various studies on susceptibility to phishing [17, 39, 45], reactions to active security warnings [26, 28, 83] and to passive security indicators [26, 51, 90].

Especially interesting is the recent study of Bluetooth pairing in presence of noise by Kaczmarek et al. [40].

Although common sense might suggest that interruptions negatively impact capacity, in the above study the likelihood of a successful task outcome increased in the presence of noise. The authors provide possible explanations for this fact based on the psychology literature. One of the explanations is that for tasks that are executed at the level much lower than the actual capacity, additional noise sharpens the attention, whereas for tasks that are executed closely to the upper bound on capacity, additional noise leads to overload and thus to lower task performance. However, as the capacity was not measured in this experiment, this assumption cannot be verified. As future work we plan to conduct realistic experiments to evaluate the proposed methodology, and to collect metrics for common security tasks.

6. A COMPLEXITY FRAMEWORK FOR USABLE SECURITY

We have seen that it is possible to measure cognitive effort and that there is scientific evidence for the existence of tasks that are beyond people's cognitive capacity. In the light of these facts, we propose a conceptual framework that attempts to clarify the definition of the term *usable security* in relation to its two dimensions, usability and security, and define several complexity categories for systems within this framework. This framework acts as a prototype structure for comparing tasks and experimental measures. It is also something of a "blunt tool" for eliminating obviously impossible tasks, where up to now the need to make such a judgement has arguably been overlooked.

Informally, a system is *secure and usable* if all tasks that the users have to execute to achieve their primary goal are within their capacity. However, this informal definition hides a lot of important details that we are able to pinpoint only in the process of rigorous reasoning about the formalisation of this concept.

6.1 Definition of the Complexity Framework

A *complexity framework for usable security* is a tuple $\langle S, U, T, SET, UET\rangle$ where:

- S is a *socio-technical system*;
- U is the specification of *intended users* of S;
- T is the set of *user tasks* that a user may be asked to execute;
- SET is the *security evaluation toolkit*, *i.e.,* the set of methods and tools used to establish S's security;
- UET is the *usability evaluation toolkit*, *i.e.,* the set of methods and processes used to establish S's usability.

In the following we describe the framework's elements in detail.

S is the specification of the socio-technical system in its operating environment. The specification includes all the system's components, their actions and interactions with the environment and with the user, including users' actions (*e.g.,* see [29]). The specification also makes *assumptions* about S analysis. These assumptions may also consider the users, such as their honesty or compliance with policies.

U is the specification of the intended users. It may include characteristics (*e.g.,* age, gender, education, or experience) that determine the users' capacities in reference to the environment where S and U operate. As discussed in Section 1, U does not necessarily model a person, but can equally represent a group of people or a person using devices that increase the ability to perform tasks.

T is the set of user tasks, such as recalling a password, deciding whether a self-signed certificate is trustworthy or understanding a policy. assessing a risk. the tasks depends on the level at which the usability should be assessed and on the available measurement methods. Thus, we may decide not to consider the task "recall a single character of a password", either because we cannot measure it, or because we are interested only in the task of recalling the whole password.

T^* denotes the set of *user traces* consisting of T and of all sequences of tasks, including the empty sequence.

SET is a tuple $\langle A, R, \texttt{AttackTraces}(S, A, R)\rangle$ where:

- A, R are the elements necessary to establish S's security. A is the model of the *adversary*, and R are the *security requirements*, *i.e.,* the desired security properties of the system.
  R can include, for instance, classical properties such as data integrity and user authentication, but also others such as verifiability [24]. A can have socio-technical capabilities (such as social engineering) and therefore exploit vulnerabilities in any component of S, including the human interfaces and human interactions.
- $\texttt{AttackTraces}(S, A, R) \subseteq T^*$ is a predicate that decides whether S, under the assumptions stated, satisfies the security requirements R in the presence of the adversary A. If S is insecure, $\texttt{AttackTraces}(S, A, R)$ returns the set of user traces with possible attacks. (*i.e.,* the *attack user traces*). If S is secure, $\texttt{AttackTraces}(S, A, R)$ returns the empty set.

We do not specify how to evaluate $\texttt{AttackTraces}(S, A, R)$, *i.e.,* how to establish the security properties of S, as this goes beyond the level of abstraction of this framework. This predicate represents the ability to say: "there is evidence that S satisfies the requirements R in the presence of the adversary A" or "there is evidence that S does not satisfy R". The evidence may be justified by applying a formal analysis method such as model checking [7, 18], or be grounded in security assessment strategies and compliance to standards, or rely on the practically established resilience to attacks ("this system is considered secure because up to now nobody has been able to break it").

UET is a tuple $\langle \texttt{Traces}(S, A), \texttt{InCapacity}(U), \texttt{OutCapacity}(U)\rangle$ where:

- $\texttt{Traces}(S, A) \subseteq T^*$ are sequences of user traces that S may ask the user to execute, including those caused by the adversary A. That is, $\texttt{AttackTraces}(S, A, R) \subseteq \texttt{Traces}(S, A)$.
  Within $\texttt{Traces}(S, A)$ we identify $\texttt{GoalTraces}(S, A)$, the set of traces that contain at least one user's *goal-reaching task*. The system's goal can be different from the user's goal. A user may aim at sending an email, the system to ensure that the e-mail is encrypted.
- $\texttt{InCapacity}(U) \subseteq T^*$ are sequences of tasks within U's capacity;
- $\texttt{OutCapacity}(U) \subseteq T^*$ are sequences of tasks beyond U's capacity.
- $\texttt{InCapacity}(U)$ and $\texttt{OutCapacity}(U)$ must satisfy the following conditions:
  (a) $\texttt{InCapacity}(U) \cap \texttt{OutCapacity}(U) = \emptyset$ *i.e.,* there is no ambiguity about which traces are within U's capacity, and which traces are beyond U's capacity;
  (b) if $\mathtt{t} \in \texttt{OutCapacity}(U)$ then $\mathtt{tt'} \in \texttt{OutCapacity}(U)$ $\forall$ $\mathtt{t'} \in T^*$ *i.e.,* if $\mathtt{t}$ is beyond the user's capacity, so it must be any extension of $\mathtt{t}$.

The two above sets define U's capabilities. The framework assumes that such capabilities can be measured, but it abstracts from any specific measurement method. In Section 5 we presented one possible measurement method and discussed others.

$\texttt{Traces}(S, A) \setminus \big(\texttt{InCapacity}(U) \cup \texttt{OutCapacity}(U)\big)$ are the user traces about which it is unknown whether they are within or beyond U's capacity.

6.2 Complexity Categories for Usable Security

In the following we propose a preliminary classification for systems depending on their usability and security properties.

6.2.1 *Usable and Unusable Security*

The role of usability for secure systems is to provide evidence that the user traces of the system are within users' capacity. Otherwise, the system cannot be claimed to be *secure and usable*. The first complexity category defines secure systems where all user traces are within users' capacity.

Definition 1 (Secure & Usable (SU))**.** *S is secure and usable if* $\texttt{AttackTraces}(S, A, R) = \emptyset$ *and* $\texttt{Traces}(S, A) \subseteq \texttt{InCapacity}(U)$.

We can also consider systems where not all, but only some traces are within U's capacity. Other traces can be beyond capacity or of unknown capacity. For this class of systems, one would either gather more evidence to prove that all their traces are within capacity or attempt to convert the system into the system S' that, with the same security guarantees, offers only traces that are known to be within the users' capacity. This is the category of systems that are partially usable.

Definition 2 (Secure & Partially Usable (SpU)).
S is secure and partially usable if $\texttt{AttackTraces}(S, A, R) = \emptyset$ *and* $\texttt{GoalTraces}(S, A) \cap \texttt{InCapacity}(U) \neq \emptyset$.

The definition above demands that there is at least one usable user's goal-reaching trace in the system. This restriction excludes from the definition systems whose users stop before reaching their gaols because of some task beyond capacity on the way.

The next complexity category defines systems that have no traces that are known to be beyond U's capacity. However, it may be unknown whether any traces within capacity exist in the system.

Definition 3 (Secure & Maybe Usable (SmU)).
S secure and maybe usable if $\texttt{AttackTraces}(S, A, R) = \emptyset$ *and* $\texttt{Traces}(S, A) \cap \texttt{OutCapacity}(U) = \emptyset$.

The complexity categories SmU and SpU both include SU. Systems in SmU but not in SpU are systems for which one would have to gather more evidence in order to understand their usability. For the next class of systems, we do not know whether they have traces that are known to be within or beyond capacity.

Definition 4 (Secure & Unknown Usable (SuU)).
S is secure and unknown usable if $\texttt{AttackTraces}(S, A, R) = \emptyset$ *and* $\texttt{Traces}(S, A) \cap \big(\texttt{InCapacity}(U) \cup \texttt{OutCapacity}(U)\big) = \emptyset$.

The next two complexity categories include systems that are secure, but not usable.

Definition 5 (Secure & Unusable ($S\neg U$)).
S is secure unusable if $\texttt{AttackTraces}(S, A, R) = \emptyset$ *and* $\texttt{GoalTraces}(S, A) \subseteq \texttt{OutCapacity}$.

Definition 5 identifies the complexity class of systems whose goal-reaching user traces are all beyond U's capacity. With respect to usability these are inappropriately designed systems.

We next define systems where only some goal-reaching traces are beyond capacity. These systems can be made usable if the unusable traces could be excluded from them through re-design, but it is not known whether the possibility for an alternative, usable design exists.

Definition 6 (Secure & Partially Unusable ($Sp\neg U$)).
S is secure and partially unusable if $\texttt{AttackTraces}(S, A, R) = \emptyset$ *and* $\texttt{GoalTraces}(S, A) \cap \texttt{OutCapacity} \neq \emptyset$.

6.2.2 Usable and Unusable Insecurity

We now define complexity categories for systems which are insecure. Here, if the usability assessment provides evidence that some attack traces are within users' capacity, the system should be considered insecure. On the contrary, if all attack traces prove to be beyond users' capacity, the system may become secure after a reassessment of its assumptions on U.

The first complexity category comprises the systems that are insecure and whose traces are all within capacity; in particular, the attack traces are also within the capacity of U. Such systems may have potentially serious security implications, since the users are able to execute all traces, including the traces that lead to an attack.

Definition 7 (Insecure & Usable ($(\neg S)U$)).
S is insecure and usable if $\texttt{AttackTraces}(S, A, R) \neq \emptyset$ *and* $\texttt{Traces}(S, A) \subseteq \texttt{InCapacity}(U)$.

The second category identifies systems that are insecure and for which there is evidence that some attack traces are usable. Therefore, the avenues for usability within the system can potentially be a catalyst (albeit indirectly) for an attack.

Definition 8 (Insecure & Partially Usable ($(\neg S)pU$)).
S is insecure and partially usable if $\texttt{AttackTraces}(S, A, R) \neq \emptyset$ *and* $\texttt{AttackTraces}(S, A, R) \cap \texttt{InCapacity}(U) \neq \emptyset$.

The next two categories define insecure systems that do not have usable attack traces, but it is also not known whether all attack traces are unusable.

Definition 9 (Insecure & Maybe Usable ($(\neg S)mU$)).
S is insecure and maybe usable if $\texttt{AttackTraces}(S, A, R) \neq \emptyset$ *and* $\texttt{AttackTraces}(S, A, R) \cap \texttt{OutCapacity}(U) = \emptyset$.

Definition 10 (Insecure & Unknown Usable ($(\neg S)uU$)).
S is insecure and unknown usable if $\texttt{AttackTraces}(S, A, R) \neq \emptyset$ *and* $\texttt{AttackTraces}(S, A, R) \cap \big(\texttt{OutCapacity}(U) \cup \texttt{InCapacity}(U)\big) = \emptyset$.

The last category represents systems that are insecure and unusable. All attack traces are beyond U's capacity, which means that this system could potentially be made secure through re-design. This can happen if the redesigned system incorporates more realistic assumptions about its users, thus eliminating attack traces that are beyond capacity. Of course, this measure alone would not guarantee that the system is *secure and usable*, as the usability and the security of the remaining traces will have to be assessed.

Definition 11 (Insecure Unusable ($\neg S \neg U$)).
S is insecure unusable if $\texttt{AttackTraces}(S, A, R) \neq \emptyset$ *and* $\texttt{AttackTraces}(S, A, R) \subseteq \texttt{OutCapacity}(U)$.

More complexity categories can be defined as needed. In the next section, we give an example on the framework usage.

6.3 Using the Framework: An Example

Let us consider a user who logs into his email account (*e.g.,* Gmail) and two procedures of authentication: the first requires the user to type an alphanumeric user-name and password; the other expects a two-step verification where the username and password are followed by a further factor stored in a USB security-key device, like the YubiKey (see www.yubico.com), activated by pressing a button on the device. The framework will help us categorise the usable security of the two procedures. We need to specify S, U, and T, and the toolkits SET and UET and so the threat model, the security requirements, and the tasks within/beyond capacity.

S includes the protocols of both the browser and Google, that is, the single and the two-step authentication procedures including the interactions with the user. The first procedure's protocol is therefore made of actions/tasks such as, for instance, "wait for the user to type the credentials", and "check the credential". The protocol either grants access or shows a warning and starts over again. insert the USB token" and "wait for the user to press the button on it". This triggers the execution of another protocol that grants access only if the security-key comes from a YubiKey previously registered to the account. S's context is limited to one mail account and one USB stick: we do not consider multiple accounts, several usernames and passwords, and plentiful USB devices.

U is a user who wants to access the mail account. We assume he can use a written paper note of the password in case he needs to recall it. In the second procedure, U includes also the YubiKey. U defines the capacities we are going to consider in the analysis.

T are the user tasks. For instance: "recalling the password", or alternatively, "retrieve the password" (*i.e.,* "read it from the paper note), "type the username and the password", "insert the USB token", and so on. The choice of the tasks determines the level of abstraction considered in the framework.

Then the framework assumes two toolkits, one for the security analysis (SET), the other for the usability analysis (UET).

SET defines the adversary (A) and the security requirements (R). We assume A to be a person who can read by shoulder surfing and memorise a password if this is written on paper and shown openly. A can also spy what is typed on a keyboard, but he is able to memorise only easy-to-recall sequences (*e.g.,* a weak password). As a security requirement we have: "the adversary should be able to impersonate the user and get access to his account". The choice of A and R defines the boundaries of our security analysis.

We have not conducted the analysis in reality but, it seems sound to imagine an attack in the first procedure, where A shoulder surfs the username and password and reuses it. This happens when the user is either retrieving a long hard-to-recall password from the paper note or entering an easy-to-recall password. In both cases, R is violated. The second scenario is more secure, at least with respect to this A. Even if A can learn the username and password pair, he cannot see the password stored in the USB device. An A with different abilities (*e.g.,* stealing objects) could still violate R, but this means changing the instantiation of the framework and performing another analysis.

UET defines the sequences of user tasks compatible with the execution of the two procedures. It also defines the goal traces, those ending with the user accessing the account, and specifies the tasks or sequences of tasks which are within/beyond U's capacity. For sake of example, we suppose to have such knowledge and we assume that "recalling the password" is the only task beyond human capacity if the password is long and random.

In the first procedure the goal can be reached only either when the user either retrieves a hard-to-recall password from the paper note and types it in, or when he recalls and types an easy-to-recall password. Such actions are within U's capacity and appear in an attack trace. Other traces, like those that include recalling a hard-to-recall password and typing it in, are possible but not within capacity. The first procedure is therefore insecure and partially usable (*i.e.,* $(\neg S)pU$). Note that if we had assumed only hard-to-recall passwords and forced U to rely on his memory to retrieve the password, all goal traces would have been beyond capacity and the adversary would have had nothing to spy upon. The procedure would have been secure and unusable (*i.e.,* $S\neg U$). In the second procedure all traces that reach the user goal are made of tasks within capacity. This time they include actions like "insert USB device" and "press button on the USB". The adversary can learn username and password, but he cannot see the second factor of authentication. Under the specific assumptions of this instance of the framework, the second procedure is secure and usable (*i.e.,* SU).

Note that we are not allowed to extend our conclusions to other procedures. In principle, we cannot conclude anything if we change the context (*e.g.,* the user has to handle multiple accounts), or our assumptions (*e.g.,* the adversary can steal objects). However, this approach be used to explore options to be both secure and productive (i.e., through choice of a particular security control, or some other support mechanism).

6.4 Improving Usable Security

Our framework suggests a partial order $>_s$ on systems according to their security. Informally speaking, $S >_s S'$ if the maximal set of security requirements that S satisfies contains the maximal set of security requirements that S' satisfies, assuming the analysis is done under the same assumptions and threat model. Two systems are equivalent in terms of security when they satisfy the same maximal set of security requirements.

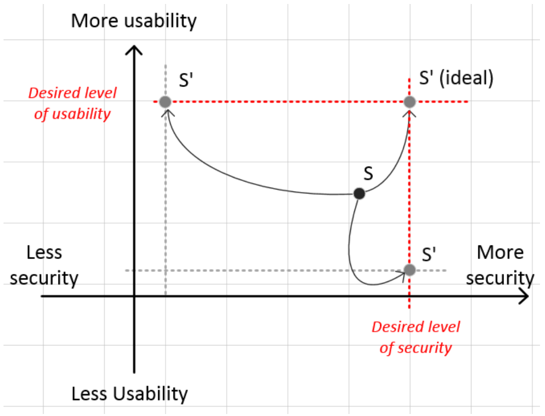

Figure 2: A representation of a situation where our framework can help. S is an actual system. S' a version of it more secure but less usable (top left), or more usable but less secure (bottom right). The ideal version (top right) improves both.

Similarly, our framework defines a partial order $>_u$ on systems according to their usability. This is defined by partitioning the space of all systems by using our complexity categories and the inclusiveness relations between them. For example, $SU >_u SpU$, because all systems in SU are at least as usable as any system in SpU, and systems in SU should be considered to be more strongly usable than those in $SpU \setminus SU$. The framework also gives rise to a usability equivalence relation.

To summarise, our framework can be used to reason about the improvement of usable security for a given system S. We may want to find a security-equivalent system specification S' that has a higher degree of usability according to the partial order among complexity categories. Otherwise we may aim to find a usability-equivalent specification with a higher degree of security. In this case we cannot identify such systems, we may ask whether one of the system's properties, security or usability, should be degraded *if* it is found to be hampering the quality of the other property. The best solution would be, however, to simultaneously improve both, security and usability, by redesigning the system. These possibilities are illustrated in Figure 2.

7. RELATED WORK

This interdisciplinary position paper intersects the areas of memory science and cognitive psychology (Section 3), ECG processing and relationship with cognitive strain (Section 5), usability (Section 2) and human-centred security. In this section we focus on discussion of related work from the latter area.

Edwards *et al.* [25] discuss the limitations of automating security for end-users and propose to study the limits of automation. Our work complements this idea, as we consider capacities and limits of the users for non-automated security mechanisms.

Cranor [22] proposes a framework for reasoning about the human in the loop when designing secure systems. The main idea is that whenever it is not possible to get humans out of the decision making process for the security mechanisms, their roles should be considered during the system design to avoid security breaches and to provide fail-safe alternatives. Our work complements this idea by providing a clear picture of what people can or cannot do.

Security ceremonies [27] model human behaviour as an integral part of security routines. As discussed in various extensions to the security ceremony analysis [6, 41, 66], security ceremonies should be expanded beyond considering humans as state machines. In our framework, we present a methodology for considering a more realistic model of human behaviour in security ceremonies by taking into account and measuring users' capacities.

Pfleeger and Caputo [62] discuss areas of behavioural science with relevance to cyber security, such as cognitive dissonance, heuristics and biases, and health behavioural models. They also examine memory limitations and where these are relevant to people's ability to perform security tasks. The authors argue that findings from behavioural sciences can be used to guide good security behaviour and reduce the perceived effort of security by providing users with appropriate tools and information. The authors propose to create a repository of such findings. We develop this idea further, proposing capture of data on the feasibility of tasks for humans, where our framework would enable objective comparison of different options in such a repository.

Work by Pfleeger et al. [63] explores other areas of behavioural sciences, such as moral values and habit formation, and how they may be leveraged to craft targeted awareness programs that can transform security behaviours. One area of focus is how to encourage better security habits, with security routines that are properly designed to support those habits. The work also refers to "user's energy", and how best to arrange tasks for the user's benefit. Our framework affords identification of security routines and mechanisms that are beneficial for the users (within capacity), avoiding those that are beyond capacity.

Becker and Beuster [5] present a methodology for formally specifying security properties of user interfaces. They augment the GOMS models with formalisms to represent an application and the assumptions that a user makes about the application, adapting concepts from human-computer interaction (HCI) to allow reasoning about the security of user interfaces, such as in e-voting systems. The work acknowledges the criticality of the user interface from a security perspective, including the impact that variations in the user interface design can have upon the capacity to reach a successful and secure outcome. We similarly provide a formal structure for reasoning about the security and usability of systems, adapting lessons from the study of cognition to understand where errors can occur. The authors posit that their framework can support automated reasoning about the potential for errors in a system. Task traces analysed under our framework may warrant development of similar capabilities.

Bonneau et al. [13] evaluate a range of technologies, such as password manager applications, one-time passwords, and hardware tokens, as alternatives to the use of text passwords. Criteria for comparison cover usability, deployability and security benefits. In this sense, the suitability of individual mechanisms is considered in terms of how they fit with the system, and the demands made upon components of a system. This includes humans, for instance whether they would have to carry anything with them or if proper use of the technology is easy to learn. In this way the work does not directly consider the capacities of the individual to complete a task, but does consider how a security mechanism may impact a user. We look at security from the perspective of the user and consider the demands that a range of tasks places upon them (where an individual may have to manage many security mechanisms).

Herley [35] uses a traditional economic approach to consider the users as exercising a rational choice over their security decisions. If a security risk is perceived as high, a user may accept a higher cost and complete the security task. Conversely, the cost of compliance may be perceived as requiring too much effort, and so a person may choose to ignore security. We consider situations where the users are not able to complete security tasks, even if they wanted to do so. In this case, one could model "beyond capacity" tasks as having infinite costs. Our framework complements this approach by providing a complexity framework for holistic evaluation of the system's security and its costs for the user.

Böhme and Grossklags consider user attention for security tasks as a public good and develop the "lump of attention" model where the consumption of user resources can be optimized from the game-theoretical perspective. In this model, if users are interrupted by security interactions, they react in an appropriate way if and only if their "attention budget" is not exhausted. This idea is close to our concept, as we propose to investigate the tasks that systematically exhaust human internal resources and therefore should not be imposed on the users at all.

Bonneau and Schechter [14] consider the capacity of human memory to remember secrets, showing that by using a mnemonic technique called "spaced repetition" people can remember 56-bit secrets with seemingly reasonable effort. Within our framework, we could consider spaced repetition as a task integrated into the authentication process and assess the security and usability of the corresponding system. Similarly, the overall suitability of other "enhancing" techniques for security tasks could be assessed using our framework.

Mannan and van Oorschot [52] show that expectations of banks on security behaviour of their online banking customers seem to be unreasonably high, as even technically savvy users do not follow them. The authors wonder whether users cannot or do not want to exhibit the required security behaviours. Our framework allows the distinction of the case where the user does not want to do a security task (then the user may be held responsible for non-compliance under some circumstances) from the cases where the user is not able to do a security task (then the user definitely should not be blamed for non-compliance).

The Compliance Budget [4] considers how the actual and perceived costs of enacting security impact upon the security behaviours of employees within organisations. Factors such as cognitive and physical load, and the individual rationalisation of both sanctions (for sidestepping security) and missed opportunities (due to abiding by security demands) are considered. These factors are framed in terms of a cost/benefit analysis, but one where an individual may have good will towards security and a willingness to behave securely for the good of the organisation, even at personal cost and lack of perceived personal benefit. The work provides heuristics for considering how security responsibilities affect the user depending upon the associated burden and disruption to primary tasks. Our work investigates how the impact of individual security mechanisms can be measured objectively.

8. CONCLUSIONS AND FUTURE WORK

In this paper we advocate the case for complexity analysis in usable security. Our claim is that usable security might not be achievable for certain tasks on account of the intrinsic characteristics of human cognitive capacities, that is, some tasks (including some advocated today) may be simply impossible for the users. We propose to systematically investigate which security tasks are *beyond capacity* for their intended users.

Understanding usable security from a complexity point of view and measuring human capacity for security tasks is invaluable for researchers and practitioners while designing security mechanisms and policies. If human capacity for a task is lower than the (overstated) security requirements, complementary security mechanisms should be designed and employed.

We develop an initial formalisation framework for complexity categories for usable security. Our taxonomy distinguishes between systems that are secure and usable (*i.e.,* within human capacity), and unusable secure systems, as well as secure systems of unknown usability. We also consider insecure systems and their relationship with usability. We identify as future work the derivation of a more comprehensive taxonomy (including principles from the behavioural sciences [62]), and investigation of relationships between systems belonging to different complexity categories. This is particularly useful for comparing the usable security of different candidate implementations of a specific system.

The implementation of a complexity framework, a future step for the authors, will generate a catalogue of average upper bounds for different implementations of common tasks and also how they vary across the different characteristics of the intended users, such as age group or expertise. This catalogue would provide rich information for the design of security mechanisms and for development and implementation of realistic security policies.

To understand the role of expertise and motivation, we will consider the influence of training and learning upon human capacities for security tasks. Work by Reason [68] has for instance examined human behaviour in the realm of safety, including different kinds of mistakes (rule-based and knowledge-based), and applications of cognitive ability to the completion of tasks (conscious, mixed, and automatic). Within our framework these findings may be useful to determine where training can be targeted to specifically support tasks, where it appears too burdensome, or where learning specific techniques itself may be beyond the capacity of intended users. In the latter cases, support mechanisms may be necessary – instead of training – to help users to complete a task.

Some tasks may strain users to their upper bound of capacity only when tasks are combined. To combine the costs of individual tasks, a unit of measurement is necessary. Future work will consider how to compose the security and usability measures of disparate tasks together, and how to objectively compare one demand with another. We hope that research methods and results from the discipline of cognitive engineering will be helpful. It considers methods for cognitive task analysis, including task switching and multitasking, which is highly relevant for the security domain [50, 70, 88].

9. ACKNOWLEDGEMENTS

We thank Ben Edwards for guidance in writing the pre-proceeding version of the paper, the NSPW 2015 anonymous reviewers and the NSPW attendees for valuable feedback, and the organisers and participants of the Dagstuhl Seminar 14491 "Socio-Technical Security Metrics" for giving us the opportunity to start this discussion.

Zinaida Benenson is supported by the Bavarian State Ministry of Education, Science and the Arts as part of the FORSEC research association. Gabriele Lenzini is supported by FNR, CORE project C11/IS/1183245 "Socio-Technical Analysis of Security and Trust", and by the European Union 7th Framework Programme (FP7/2007-2013) under grant agreement no. 318003 (TRESPASS). Daniela Oliveira is supported by the National Science Foundation grants CNS-1464801 and SES-1450624. Simon Parkin is supported by UK EPSRC, grant nr. EP/K006517/1 ("Productive Security"). Sven Uebelacker is supported by the European Union 7th Framework Programme (FP7/2007-2013) under grant agreement no. 318003 (TRESPASS).

10. REFERENCES

[1] P. O. Astrand, , and K. Rodahl. *Textbook of Work Physiology: Physiological Bases of Exercise*. McGraw-Hill, 2003.

[2] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on*, 1(1):11–33, 2004.

[3] A. Beautement, R. Coles, J. Griffin, C. Ioannidis, B. Monahan, D. Pym, A. Sasse, and M. Wonham. Modelling the human and technological costs and benefits of usb memory stick security. In *Managing Information Risk and the Economics of Security*, pages 141–163. Springer, 2009.

[4] A. Beautement, M. A. Sasse, and M. Wonham. The compliance budget: managing security behaviour in organisations. In *Proceedings of the 2008 workshop on New security paradigms*, pages 47–58. ACM, 2009.

[5] B. Beckert and G. Beuster. A method for formalizing, analyzing, and verifying secure user interfaces. In *Formal methods and software engineering*, pages 55–73. Springer, 2006.

[6] G. Bella and L. Coles-Kemp. Seeing the full picture: the case for extending security ceremony analysis. 2011.

[7] G. Bella, R. Giustolisi, and G. Lenzini. Socio-Technical Formal Analysis of TLS Certificate Validation in Modern Browsers. In *Proc. of PST 2013*. IFIP, 2013.

[8] Z. Benenson, A. Girard, N. Hintz, and A. Luder. Susceptibility to URL-based Internet attacks: Facebook vs. email. In *6th IEEE International Workshop on SEcurity and SOCial Networking (SESOC)*, pages 604–609. IEEE, 2014.

[9] The biosig project http://biosig.sourceforge.net/.

[10] J. Blocki. *Usable Human Authentication: A Quantitative Treatment*. PhD thesis, Carnegie Mellon University Pittsburgh, PA, 2014.

[11] P. Bongers, C. de Winter, M. Kompier, and V. Hildebrandt. Psychosocial Factors at Work and Musculoskeletal Disease. *Scand J Work Environ Health*, 19(5):297–312, 1993.

[12] P. Bongers, C. Winter, M. Kompier, and V. Hildebrandt. Distributed Cognition: Toward a New Foundation for Human-Computer Interaction Research. *ACM Transactions on Computer-Human Interaction*, 7(2):174–196, 2000.

[13] J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 553–567. IEEE, 2012.

[14] J. Bonneau and S. Schechter. Towards reliable storage of 56-bit secrets in human memory. In *Proc. USENIX Security*, 2014.

[15] S. Brostoff and M. A. Sasse. Safe and sound: a safety-critical approach to security. In *Proceedings of the 2001 workshop on New security paradigms*, pages 41–50. ACM, 2001.

[16] J. S. Brown, A. Collins, and P. Duguid. Situated Cognition and the Culture of Learning. *Educational Researcher*, 18(1):174–196, 1989.

[17] D. Caputo, S. Pfleeger, J. Freeman, and M. Johnson. Going spear phishing: Exploring embedded training and awareness. *IEEE Security & Privacy*, 12(1):28–38, 2014.

[18] E. M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT press, 1999.

[19] P. L. Clemens. Human factors and operator errors. 2002.

[20] E. Cooper-Martin. Pupillary responses index cognitive resource limitations. *Marketing Letters*, 5(1):43–56, 1994.

[21] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.

[22] L. F. Cranor. A framework for reasoning about the human in the loop. In *Proceedings of the 1st Conference on Usability, Psychology, and Security*, number 1 in UPSEC'08, pages 1:1–1:15, 2008.

[23] L. F. Cranor and S. Garfinkel, editors. *Security and usability: designing secure systems that people can use*. O'Reilly Media, Inc., 2005.

[24] J. Dreier, R. Giustolisi, A. Kassem, P. Lafourcade, and G. Lenzini. A framework for analyzing verifiability in traditional and electronic exams. In *Proc. of the Information Security Practice and Experience - 11th International Conference, ISPEC 2015, Beijing, China, May 5-8, 2015*, volume 9065 of *Lecture Notes in Computer Science*, pages 514–529. Springer, 2015.

[25] W. K. Edwards, E. S. Poole, and J. Stoll. Security Automation Considered Harmful? In *Proceedings of the 2007 Workshop on New Security Paradigms*, NSPW '07, pages 33–42, 2008.

[26] S. Egelman and L. F. Cranor. You've Been Warned: An Empirical Study of the Effectiveness of Web Browser Phishing Warnings. *ACM Conference on Human Factors in Computer Systems (CHI)*, 2008.

[27] C. M. Ellison. Ceremony design and analysis. *IACR Cryptology ePrint Archive*, 2007.

[28] A. P. Felt, R. W. Reeder, H. Almuhimedi, and S. Consolvo. Experimenting at scale with Google chrome's SSL warning. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 2667–2670. ACM, 2014.

[29] A. Ferreira, J.-L. Huynen, V. Koenig, and G. Lenzini. A Conceptual Framework to Study Socio-Technical Security. In T. Tryfonas and I. Askoxylakis, editors, *Human Aspects of Information Security, Privacy, and Trust*, volume 8533 of *Lecture Notes in Computer Science*, pages 318–329. Springer International Publishing, 2014.

[30] D. Florêncio, C. Herley, and P. C. Van Oorschot. Password portfolios and the finite-effort user: Sustainably managing large numbers of accounts. In *Proc. USENIX Security*, 2014.

[31] M. R. Garey and D. S. Jonhson. *Computers and Intractability: A guide to the theory of NP-Completeness*. W. H. Freeman & Co., 1979.

[32] S. Garfinkel and H. R. Lipford. *Usable Security: History, Themes, and Challenges*. Morgan & Claypool Publishers, 2014.

[33] S. L. Garfinkel and R. C. Miller. Johnny 2: a user test of key continuity management with s/mime and outlook express. In *Proceedings of the 2005 symposium on Usable privacy and security*, pages 13–24. ACM, 2005.

[34] E. Granholm, S. K. Morris, A. J. Sarkin, R. F. Asarnow, and D. V. Jeste. Pupillary responses index cognitive resource limitations. *Psychophysiology*, 33(3):457–461, 1996.

[35] C. Herley. So Long, and No Thanks for the Externalities: the Rational Rejection of Security Advice by Users. *New Security Paradigms Workshop (NSPW)*, 2009.

[36] C. Herley. More is not the answer. *IEEE Security & Privacy*, 12(1):14–19, 2014.

[37] N. Hjortscov, D. Rissen, A. K. Blansted, N. Fallentin, U. Lundberg, and K. Sogaard. The Effect of Mental Stress on Heart Rate Variability and Blood Pressure During Computer Work. *European Journal of Applied Physiology*, 92:84–89, 2004.

[38] ISO 9241-11:1998 Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability, 2000.

[39] T. N. Jagatic, N. A. Johnson, M. Jakobsson, and F. Menczer. Social Phishing. *Communications of ACM*, 50(10), 2007.

[40] T. Kaczmarek, A. Kobsa, R. Sy, and G. Tsudik. An unattended study of users performing security critical tasks under adversarial noise. In *USEC*, 2015.

[41] C. Karlof, J. D. Tygar, and D. Wagner. Conditioned-safe ceremonies and a user study of an application to web authentication. In *NDSS*, 2009.

[42] A. Kerckhoffs. La Cryptographie Militaire. *Journal des sciences militaires*, IX(6):5–38, 1883.

[43] I. Kirlappos, S. Parkin, and M. A. Sasse. Learning from "shadow security": Why understanding non-compliance provides the basis for effective security. In *USEC*, 2014.

[44] S. Komanduri, R. Shay, L. F. Cranor, C. Herley, and S. Schechter. Telepathwords: Preventing weak passwords by reading users' minds. In *23rd USENIX Security Symposium (USENIX Security 14). San Diego, CA: USENIX Association*, pages 591–606, 2014.

[45] P. Kumaraguru, S. Sheng, A. Acquisti, L. F. Cranor, and J. Hong. Teaching johnny not to fall for phish. *ACM Transactions on Internet Technology (TOIT)*, 10(2):7, 2010.

[46] C. Kuo, A. Perrig, and J. Walker. Designing an evaluation method for security user interfaces: lessons from studying secure wireless network configuration. *interactions*, 13(3):28–31, 2006.

[47] K. Kuutti. *Context and consciousness (Chapter 2 - Activity Theory as a Potential Framework for Human-Computer Interaction Research)*. The MIT Press, 1995.

[48] B. Lampson. Privacy and security usable security: how to get it. *Communications of the ACM*, 52(11):25–27, 2009.

[49] J. Lazar, J. H. Feng, and H. Hochheiser. *Research Methods in Human-Computer Interaction*. Willey, 2010.

[50] J. D. Lee, A. Kirlik, and M. J. Dainoff. *The Oxford handbook of cognitive engineering*. Oxford University Press, 2013.

[51] E. Lin, S. Greenberg, E. Trotter, D. Ma, and J. Aycock. Does domain highlighting help people identify phishing sites? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2075–2084. ACM, 2011.

[52] M. Mannan and P. C. van Oorschot. Security and usability: The gap in real-world online banking. In *Proceedings of the 2007 Workshop on New Security Paradigms*, NSPW '07, pages 1–14, 2008.

[53] M. Mather. *When I'm 64 - A Review of Decision-Making Processes: Weighing the Risks and Benefits of Aging*. The National Academies Press, 2006.

[54] D. McCarney, D. Barrera, J. Clark, S. Chiasson, and P. C. van Oorschot. Tapas: design, implementation, and usability evaluation of a password manager. In *Proceedings of the 28th Annual Computer Security Applications Conference*, pages 89–98. ACM, 2012.

[55] G. A. Miller. The magical number seven, plus or minus two. *Psychological Review*, 63(2):81–97, 1956.

[56] J. Nielsen. *Usability engineering*. Elsevier, 1994.

[57] D. Oliveira, M. Rosenthal, N. Morin, M. K.-C. Yeh, J. Cappos, and Y. Zhuang. It's the Psychology Stupid: How Heuristics Explain Software Vulnerabilities and How Priming can Illuminate Developer's Blind Spots. *Annual Computer Security Applications Conference (ACSAC)*, 2014.

[58] F. Paas and J. Van Merriënboer. The Efficiency of Instructional Conditions: An Approach to Combine Mental Effort and Performance Measures. *Human Factors*, 35(4):737–743, 1993.

[59] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley Longman, 1994.

[60] S. A. Paritala. Effects of Physical and Mental Tasks on Heart Rate Variability. Master Thesis, Louisiana State University, 2009.

[61] S. E. Parkin, R. Y. Kassab, and A. Van Moorsel. *The impact of unavailability on the effectiveness of enterprise information security technologies*. Springer, 2008.

[62] S. L. Pfleeger and D. D. Caputo. Leveraging behavioral science to mitigate cyber security risk. *Comput. Secur.*, 31(4):597–611, June 2012.

[63] S. L. Pfleeger, M. A. Sasse, and A. Furnham. From weakest link to security hero: Transforming staff security behavior. *Journal of Homeland Security and Emergency Management*, 11(4):489–510, 2014.

[64] Polar rs800 http: //www.polar.com/us-en/products/earlier_products/RS800.

[65] I. Pollack. The information of elementary auditory displays. *J. Acoust. Soc. Amer.*, 24:745–749, 1952.

[66] K. Radke, C. Boyd, J. G. Nieto, and M. Brereton. Ceremony analysis: Strengths and weaknesses. In *Future Challenges in Security and Privacy for Academia and Industry*, pages 104–115. Springer, 2011.

[67] J. Reason. *Human error*. Cambridge university press, 1990.

[68] J. Reason. *The human contribution*. Ashgate Burlington, VT, 2008.

[69] K. Renaud, M. Volkamer, and A. Renkema-Padmos. Why Doesn't Jane Protect Her Privacy? In *Privacy Enhancing Technologies*, pages 244–262. Springer, 2014.

[70] E. M. Roth, E. S. Patterson, and R. J. Mumaw. Cognitive engineering. *Encyclopedia of software engineering*, 2002.

[71] S. Ruoti, N. Kim, B. Burgon, T. Van Der Horst, and K. Seamons. Confused johnny: when automatic encryption leads to confusion and mistakes. In *Proceedings of the Ninth Symposium on Usable Privacy and Security*, page 5. ACM, 2013.

[72] J. Saltzer and M. Schroeder. The Protection of Information in Computer Systems. *Proceedings of the IEEE*, 63(9):338–402, 1975.

[73] M. Sasse, M. Steves, K. Krol, and D. Chisnell. The great authentication fatigue – and how to overcome it. In P. Rau, editor, *Cross-Cultural Design*, volume 8528 of *Lecture Notes in Computer Science*, pages 228–239. Springer International Publishing, 2014.

[74] M. A. Sasse and I. Flechais. Usable security: Why do we need it? how do we get it? In L. F. Cranor and S. Garfinkel, editors, *Security and Usability: Designing secure systems that people can use*. O'Reilly, 2005.

[75] D. L. Schacter. The Seven Sins of Memory: Insights from Psychology and Cognitive NeuroScience. *American Psychologist*, 54(3):182–203, 1999.

[76] D. L. Schacter. *The Seven Sins of Memory: How the Mind Forgets and Remembers*. Houghton Mifflin Company, 2001.

[77] Scientific American Editors. *Remember When?: The Science of Memory*. Scientific American, 2013.

[78] C. E. Shannon and W. Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, Urbana, Illinois, 1949.

[79] R. Shay, S. Komanduri, A. L. Durity, P. S. Huh, M. L. Mazurek, S. M. Segreti, B. Ur, L. Bauer, N. Christin, and L. F. Cranor. Can long passwords be secure and usable? In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 2927–2936. ACM, 2014.

[80] A. Shostack and A. Stewart. *The new school of information security*. Pearson Education, 2008.

[81] D. K. Smetters and R. E. Grinter. Moving from the design of usable security technologies to the design of useful secure applications. In *Proceedings of the 2002 workshop on New security paradigms*, pages 82–89. ACM, 2002.

[82] M. Steves, D. Chisnell, A. Sasse, K. Krol, M. Theofanos, and H. Wald. Report: Authentication diary study. 2014.

[83] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor. Crying Wolf: An Empirical Study of SSL Warning Effectiveness. *ACM Conference on Human Factors in Computer Systems (CHI)*, 2008.

[84] P. Taraveinen, A. Karjaleinen, and O. Ranta-aho. An Advanced Detrending Method with Application to HRV Analysis. *IEEE Trans Biomed Engineering*, 49(2):172–175, 2001.

[85] M. F. Theofanos and S. L. Pfleeger. Guest editors' introduction: Shouldn't all security be usable? *IEEE Security & Privacy*, (2):12–17, 2011.

[86] D. M. Wegner. Transactive memory: A contemporary analysis of the group mind. In *Theories of group behavior*, pages 185–208. Springer, 1987.

[87] A. Whitten and J. D. Tygar. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *Usenix Security*, 1999.

[88] C. D. Wickens, S. E. Gordon, and Y. Liu. *An introduction to human factors engineering*. Pearson Prentice Hall Upper Saddle River, NJ:, 2004.

[89] C. D. Wickens, J. G. Hollands, S. Banbury, and R. Parasuraman. *Engineering psychology and human performance*. Pearson, 2013.

[90] 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*, pages 601–610. ACM, 2006.

[91] M. E. Zurko. User-centered security: Stepping up to the grand challenge. In *Computer Security Applications Conference, 21st Annual*, pages 14–pp. IEEE, 2005.

[92] M. E. Zurko and R. T. Simon. User-centered security. In *Proceedings of the 1996 workshop on New security paradigms*, pages 27–33. ACM, 1996.