

# Introducing Usability to the Common Criteria

Luke Church<sup>1</sup>, Matthew Nicolas Kreeger<sup>2</sup>, and Marcus Streets<sup>2</sup>

<sup>1</sup> University of Cambridge, Computer Laboratory, Cambridge, England, U.K.

luke@church.name

<sup>2</sup> nCipher Corporation, Cambridge, England, U.K.

{matthew, marcus}@ncipher.com

**Abstract.** In this paper we argue that Common Criteria evaluations need to take usability into account and make recommendations about how this might be achieved.

We argue that security and usability are entangled in a way that means that evaluating security without evaluating usability is both philosophically and practically flawed. To illustrate this we describe a number of cases in which usability failings have resulted in security failures and conclude that evaluating usability is vital to ensuring security.

We then examine some of the consequences of the low emphasis on usability in the Common Criteria and argue that it results in a problematic incentive for developers to tradeoff usability for “theoretical security” which decreases the effective security of the deployed system. Further, we argue that the lack of emphasis on usability fails to ensure that a certified product is actually secure. We conclude that it is important that usability be included in Common Criteria evaluations.

Finally we present a set of recommendations as to how to achieve this.

## 1 Introduction

It has become something of a truism in security engineering that “usability is a problem”. From Anderson’s paper on real world failures in cryptosystems [1], through Whitten’s analysis of PGP [2] to recent concerns over the introduction of Microsoft Vista’s User Access Control [3] and phishing related security problems [4], usability dominates many of the technical considerations in limiting the adoption, and effective security, of deployed systems.

Whilst it might be convenient to explain away such problems as user error, when the errors are as common as they appear to be, such an explanation defeats the point of the Common Criteria (CC) [5]; as we shall argue in more detail, if the purpose of the CC is to give some assurance that a system is secure, it is not sufficient to show that it is secure “in theory”. For example PGP 5.0<sup>3</sup> was reasonably secure “in theory”, yet in Whitten’s study 3 of the 12 participants accidentally revealed the secret information that they were supposed to be using PGP to protect. It seems to us that this is hardly assurance of a secure product.

---

<sup>3</sup> PGP 5.0 was not to the best of our knowledge submitted to for CC evaluation. We use it only as an example of where usability concerns rendered an otherwise fairly secure product highly problematic.

## II

By analogy, consider a scheme to assure the safety of a motor car. If a car was submitted to testing that had the positions of the gas and brake pedals exchanged, it might still pass technical assurance tests perfectly. However it is probably a dangerous car for many drivers. In order to assure potential customers of the car's safety, it is necessary to include an analysis of the interaction of people with the car.

In terms of providing product assurance Anderson [6] offers the view that technical evaluation alone is insufficient and that assessing the behaviour of the application in the real world is necessary. We believe that by evaluating the security of a system without considering one of its most important interactions (that between the system and its users) the CC provides much less assurance than it might.

Therefore, this paper discusses how and why to modify the CC to take usability into account.

## 2 Usability in the Common Criteria 3.3

Let us start by considering in more detail why usability should be included in the CC. To motivate evaluating usability in the CC, we need to show:

- It is necessary to evaluate usability to provide assurance of security
- It is necessary to perform such an evaluation within the CC

Let us take these in turn. In the introduction we made the argument that evaluating usability was necessary in order to assure the customer of security. This needs a little further unpacking, as it will shape the way in which we propose evaluation should occur.

### 2.1 Why do we need to evaluate usability?

In this section we shall argue that, for many systems, security cannot be evaluated without considering usability. Our argument consists of roughly three parts:

- Pragmatics: What actually happens to unusable systems?
- Social Interactions: How does the social use of technology affect security?
- Systemic Issues: What can we say in the abstract about security and usability?

#### Pragmatics

We observed in the introduction that there have been concerns over security usability for some time now. This gives us a collection of systems that we can use to consider the security behaviour of users in the presence of poor usability. A number of general points can be identified. Note that this list is by no means exhaustive.

### ***Unusable security systems get bypassed***

Systems that exhibit problematic usability are often bypassed. For example, if a system provides warnings to the user on a regular basis they learn to just “click ok”, without considering the content of the dialog. When the case arises where they should have clicked cancel, the muscle memory is too engrained and they make the wrong choice [3].

This tendency to avoid security that is so problematic applies even to high value cases. The Office of Inspector General [7] discusses a case where a former CIA Deputy Director bypassed the security provisions and worked on his home computer, which had also been used to access high-risk websites.

This emerges as a general theme: If security gets in the way too much, it gets ignored. Without testing for this issue, the security of a practical system may not be as strong as it appears.

### ***Human errors must be accounted for***

As the previous examples indicate, humans are not perfect information processing devices. Just as they learn a behaviour of “clicking OK”, so they have other errors. Kahneman et al [8] discuss a range of heuristics and biases which affect human decision making. These include, for example, confirmation bias: people seek evidence to support a hypothesis rather than to refute it. Such biases are important, in for example understanding when users are likely to fall for phishing attacks.

A related example is “post completion error”. When completing a multi-step process, any steps that come after the step where the user appears to have completed their goal are likely to be forgotten. Examples of user interfaces that are designed with this in mind are ATMs in the UK. They require the user to remove their card from the machine before they are given their cash. Otherwise giving the user their cash completes the task they went to the ATM to perform and the post-completion operation of collecting their card is forgotten.

There is anecdotal evidence [9] that Chip and Pin systems cause post-completion errors, this affects their security requirements, as the shop must deal with numbers of customers leaving their cards behind. Again, a discussion of the “security” of the system from the customers’ perspective without taking this kind of factor into account is misleading.

So we have seen that the way individuals interact with security technologies has a substantial effect on the security properties of the system.

### **Social Use of Technology**

Many of the previous examples considered how security technology was used by individuals, however this is in itself much too limited. The social context in which security technology is used has a substantial effect on the behaviour of the users.

To start with a simple example, consider a company where most users do not lock their workstations when they are away from a desk. The first user to do so might be considered to be either untrusting or untrustworthy by their colleagues, creating a social pressure for them not to do so. Here the social context in which the security system exists affects the way it is used. Failure to account for this social context results in a difference between the theoretical and practical security properties of the system.

The example of locking desktops is a fairly simple behavioural change caused by the social context. As security systems are frequently used to enforce some social rules, the effects of the social context often run much deeper. Consider for example the medical environment described in [10]. Users authenticate themselves to drug delivery machines to adjust the treatment levels. Such actions are audit logged.

At an individual level, it is clearly very important to ensure that users can authenticate successfully and change the drug delivery levels accurately. However, this level of analysis is insufficient. In medical environments, the nurse often changes a drug delivery level on behalf of a doctor, the nurse may not even agree with the change. However the change is audit-logged as being an action that the nurse approved. As such the security properties that the system appears to offer are notably different to the ones actually offered when the social context is accounted for.

It has been known for some time that the relationship between technology and its social use is a complex one and is often a process of negotiation between designer and user [11, 12]. The avoidance of problematic security technologies discussed earlier can be considered as a form, albeit a somewhat impoverished one, of this process of negotiation.

Security technologies often contain a model of the social context that they are used in (be it explicit like Bell-LaPadula [13] or implicit like the previous example), but there is often limited ability to negotiate this model after the technology has been deployed. Given this, it is perhaps unsurprising that failing to explicitly account for this social context can result in the theoretical security properties of the system differing very substantially from actual security properties of the system in use.

### **Systemic Issues**

We have seen above that the context in which security systems are used, both on the individual and the social level, has a substantial effect on the security properties of the whole system. In this section we argue that this dependency (security on usability) is not an accidental quirk of the systems we have considered, but rather a fundamental aspect of security engineering.

Security is a holistic property of a system. Typically technology makes up one part of the system, users and management make up another part, et cetera. The CC explicitly acknowledges this by observing that there may be non-technical controls [5, Section A.6.3]. In assuring security, the CC is trying to give confidence that the technical component of a system meets a set of properties.

However, the technical component must interact with other components in order to be useful. Again, the CC acknowledges this, with a discussion of interface assurance [5, Section ADV\_FSP].

We state that ignoring usability is to ignore the interaction between the human component(s) of the system and the technical component(s) of the system. As such no assurance of security of the whole can be provided. Attempts to provide such assurances are particularly problematic in the common case where the security system is used to ensure that the behaviour of the human elements of the system is constrained in some way.

So it is essential to consider the interaction between the system and the users of a system, in their social context, if one is to provide assurances of security.

We conclude that one cannot assure security without assuring usability.

## 2.2 Why is the CC the right place to do this?

If we accept the above argument that usability needs to be assured in order to assure security, then the remaining question is: should we do this within the CC? Why not have a separate process to assure usability?

We shall address this question from two perspectives. Firstly we shall argue that there are substantial negative effects of not addressing usability within the CC. Secondly we argue that security and usability are so intertwined that it is not practical to evaluate them separately. Hence the CC is the right place to do the evaluation.

### Effects of not evaluating Usability in the CC

By the arguments of Section 2.1, not evaluating usability within the Common Criteria prevents it being useful for making general claims about security, even with respect to a particular protection profile. It restricts the notion of “being secure” to a set of specified technical properties.

This has a number of unfortunate consequences:

- It creates a false perception as to what is being assured by a CC evaluation
- It encourages unrealistic expectations as to what a user is capable of
- It creates an unfortunate incentive to migrate security decisions to the user
- It fails to motivate progress in security usability

We now consider these issues in turn.

***False perception of what is being assured***

The lack of discussion of user behaviour in Common Criteria, except for the minimal requirements in the AGD class, gives rise to a set of fluid assumptions about what the user may or may do. In many ways this can be viewed as an extreme case of abstracting away the user. As discussed in [14], this can be seriously detrimental to usability.

Further, it is unclear how customers of CC certified products interpret the lack of discussion of usability.

***Unrealistic assumptions as to what a user is capable of***

In a related issue to the previous one, the lack of an explicit discussion about usability can result in unrealistic assumptions about what a user is capable of doing. The authors have seen one example where an assumption was made that the user had the capability of a UNIX system administrator, when they were a HR secretary.

Without an explicit discussion of what the users are and are not expected to be able to do, there is no way of understanding for whom, and in what context the product is secure.

***Migration of security decisions to user***

The lack of any explicit discussion of usability results in an incentive to move security decisions to the user. If the system is only being evaluated for technical security, and the user can be treated as an oracle, then migrating decisions from the system to the user is likely to make it easier for the developer to have the system certified. However this is likely to be substantially harmful to the usability of the system.

***Missed opportunity to motivate security usability***

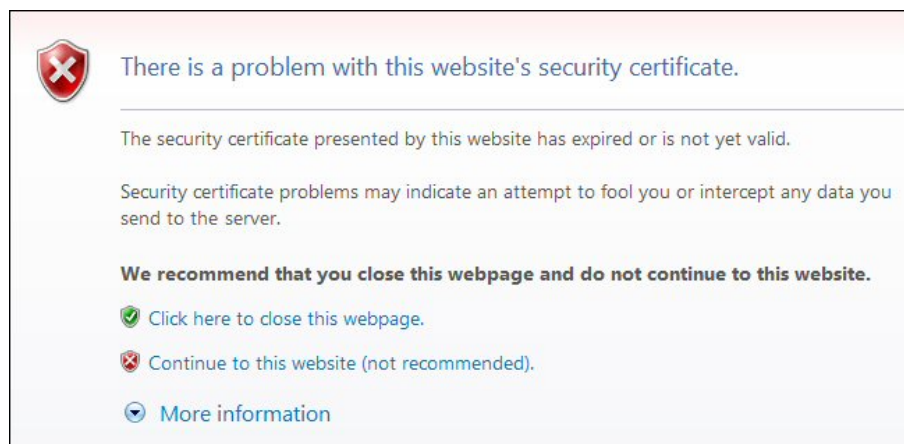
As the CC is required by a number of purchasers, requiring a usability evaluation would create a helpful incentive for product manufactures to consider usability. There is precedent for this. The introduction of Section 508 [15] accessibility requirements, whilst not perfect, at least brought awareness of accessibility issues in websites to the foreground. By not including requirements for usability, the CC misses an important opportunity.

***Intertwining of security and usability***

Finally we argue that security and usability are so intertwined that separate descriptions of them would be incoherent. For example, consider the changes in behaviour of Internet Explorer in response to a problem with an SSL certificate (see Fig. 1).

**Fig. 1.** SSL certificate: changes in problem response

(a) Internet Explorer Version 6



(b) Internet Explorer Version 7

A description of the migration from version 6 to version 7 that only considered usability, or only considered security would be strange. The change was made, in response to a problem with the usability of a security mechanism, the issues cannot be separated into two documents whilst maintaining coherence.

This difficulty would also apply to previous examples in this paper. Discussions of the usability of PGP would be very different, if there were no security implications, likewise with the hospital example detailed in Section 2.1.

We have seen many examples of where not considering usability in the design of security systems has proven to be harmful, the same is definitely true of ignoring security in the design of user interfaces. Consequently to achieve a coherent description of the system being evaluated, it is necessary to consider both in the context of each other. Hence the Common Criteria is the right place to discuss security and its usability.

Conclusion: We need to evaluate usability in the CC.

### 3 Engineering Usable Security

So from the above, we need to evaluate usability within the CC, how might we go about doing this?

Rather than provide proscriptive guidance, which would be difficult to do generally, does not fit well with the CC model and by effect of its generality would have to be limited in scope, we suggest a process whereby the description of products under evaluation are required to include a number of pieces of extra information.

#### 3.1 Proposed Changes to CC to include Usability

We suggest adding a new class to the CC to cover User Identification.

Information provided in this class would then need to be added as dependencies to all other classes, to ensure that the work done here is carried through the rest of the design process.

#### **Class AUI: User identification.**

##### **User Identification (AUI\_USR)**

###### *Objectives:*

A common failing in usability is failing to identify correctly who your users are [14]. The current user base of computer systems varies widely, from computer scientists to home users and children. It is crucial that the product is designed with its users in mind. Companies often design for the wrong user group. It is essential that this does not occur. Consequently effort needs to have been invested to determine that the believed user base is the correct one.

**AUI\_USR.1**

*Dependencies:* No dependencies.

*Developer action element:*

**AUI\_USR.1.1D** The developer shall provide a statement that identifies the class of person that is expected to use the TOE.

*Content and presentation elements:*

**AUI\_USR.1.1C** The statement shall include information on the users expected educational attainments and areas of technical competence.

**AUI\_USR.2**

*Dependencies:* AUI\_USR.1

*Developer action element:*

**AUI\_USR.2.1D** The developer shall provide evidence that supports the claims in AUI\_USR.1.

*Content and presentation elements:*

**AUI\_USR.2.1C** The evidence shall demonstrate that the vendor has taken best efforts to identify the users for the TOE.

**Social Context (AUI\_SCT)**

*Objectives:*

The social context in which security technology is used has a substantial effect on the behaviour of the users.

Failing to account for the social context in which technology is used can result in the apparent security properties of the system differing very substantially from actual security properties of the system in use.

**AUI\_SCT.1**

*Dependencies:* AUI\_USR.1

*Developer action element:*

**AUI\_SCT.1.1D** The developer shall provide a statement that identifies the social context in which the TOE is expected to be used.

**AUI\_SCT.2**

*Dependencies:* AUI\_SCT.1

*Developer action element:*

**AUI\_SCT.2.1D** The developer shall provide evidence that supports the claims in AUI\_SCT.1.

*Content and presentation elements:*

**AUI\_SCT.2.1C** The evidence shall demonstrate that the vendor has taken best efforts to identify the social context in which the TOE will be deployed.

## Usability (AUI\_USE)

### *Objectives*

The objective of this family is to ensure that the security features of the TOE can be operated correctly by the identified user in the identified social context.

Given the wide range of possible TOEs that can be submitted it is not possible to enumerate the tests that could be carried out.

It is the duty of the evaluator to determine whether the information provided by the vendor satisfies this requirement for the specific TOE under test.

### **AUI\_USE.1**

*Dependencies:* AUI\_USR.1, AUI\_SCT.1, AGD\_OPE.1, AGD\_PRE.1

*Developer action element:*

**AUI\_USE.1.1D** The developer shall provide evidence to demonstrate that the security principles and interfaces of the TOE as set out in the user guidance are understandable and usable by the identified users.

We would suggest the elements be introduced at the following Common Criteria Assurance Levels.

- EAL 1 AUI\_USR.1, AUI\_USE.1
- EAL 2 add AUI\_SCT.1
- EAL 3 add AUI\_USR.2
- EAL 4 add AUI\_SCT.2

## 4 Concluding Remarks

That security and usability are inherently entangled is widely accepted [16, 17]. However, introducing the concept of usability to security focused evaluation and assurance schemes, such as Common Criteria, has been, to some extent, overlooked [6, 18].

We have provided an overview of the consequences that may result from failing to include usability within Common Criteria; we have also attempted to persuade the reader of the necessity of its inclusion, and to propose a set of initial recommendations as to how this may be achieved through the introduction of a usability focused class.

## References

1. Anderson, R.: Why cryptosystems fail. In: CCS '93: Proceedings of the 1st ACM conference on Computer and communications security, ACM (1993) 215–227
2. Whitten, A., Tygar, J.D.: Why johnny can't encrypt: a usability evaluation of PGP 5.0. In: SSYM'99: Proceedings of the 8th conference on USENIX Security Symposium, USENIX Association (1999)

3. Schneier, B.: Microsoft Vista's endless security warnings. [http://www.schneier.com/blog/archives/2006/04/microsoft\\_vista.html](http://www.schneier.com/blog/archives/2006/04/microsoft_vista.html) (2006) Retrieved 07 11, 2008.
4. Cranor, L.F.: Supporting trust decisions. <http://cups.cs.cmu.edu/trust.php> (2007) Retrieved 07 11, 2008.
5. Common Criteria Recognition Agreement: Common Criteria for Information Technology Security Evaluation. ISO (2006)
6. Anderson, R.: Security Engineering: A Guide to Building Dependable Distributed Systems. John Wiley & Sons (2001)
7. Snider, L.B., Seikaly, D.S.: Improper handling of classified information by John M. Deutch. Technical report, 1998-0028-IG, Central Intelligence Agency, United States of America (2000)
8. Kahneman, D., Slovic, P., Tversky, A., eds.: Judgement under Uncertainty: Heuristics and Biases. Cambridge University Press (1982)
9. Church, L.: Broken social protocols: Chip and pin. <http://lukechurch.blogspot.com/2007/04/broken-social-protocols-chip-and-pin.html> (2007) Retrieved 07 11, 2008.
10. Church, L.: End user security: The democratization of security usability. (2008) Presented at SHB 2008: Interdisciplinary Workshop on Security and Human Behaviour.
11. MacKenzie, D., Wajcman, J., eds.: The Social Shaping of Technology. Open University Press (1985)
12. Bowker, G.C., Star, S.L.: Sorting Things Out: Classification and Its Consequences. MIT Press (2000)
13. Bell, D.E., LaPadula, L.J.: Secure computer systems: Mathematical foundations. Technical report, MTR-2547-I, MITRE Corporation (1973)
14. Blackwell, A.F., Church, L., Green, T.G.: The abstract is 'an enemy': Alternative perspectives to computational thinking. (2008) Submitted to the 20th Annual Workshop of The Psychology of Programming Interest Group.
15. National Archives and Records Administration: Section 508 standards. [http://www.section508.gov/final\\_text.html](http://www.section508.gov/final_text.html) (2000) Retrieved 07 11, 2008.
16. Saltzer, J.H., Schroeder, M.D.: The protection of information in computer systems. Proceedings of the IEEE **63**(9) (1975) 1278–1308
17. Cranor, L.F., Garfinkel, S., eds.: Security and Usability: Designing Secure Systems that People Can Use. O'Reilly (2005)
18. Grimm, R., Krimmer, R., Meißner, N., Reinhard, K., Volkamer, M., Weinand, M., Helback, J.: Security requirements for non-political internet voting. Technical report, Nr. 06/2007, Universität Koblenz-Landau, Germany (2007)