Developing a User Interface Security Assessment Method

Sarah Abdellahi and Heather Richter Lipford, *University of North Carolina at Charlotte*
Carrie Gates and Jeremiah Fellows, *Bank of America*

Abstract

Human behavior can play a major role in many security breaches and issues within organizations. While there are a variety of methods to assess the usability of interfaces, and the security risks and vulnerabilities of software, there are few methods at the intersection. We propose the User Interface Security Assessment (UISA) method to allow application designers and developers to assess the security risks that result from user interaction with a particular application interface. UISA can help stakeholders to understand the implications of specific design decisions and identify areas for design and security improvements.

1. Introduction

Most software designs, whether for external customers or internal employees, are evaluated based on the expected usability and user experience of the application. Many organizations have mature processes for assessing whether a design is intuitive, efficient to use, effective, and in line with the design patterns of the organization. Similarly, many applications with security implications can be evaluated by an application security team for vulnerabilities introduced by the software architecture or implementation. Yet, as the usable security community has long established, user behaviors and decisions play an important part in the overall security of an application, and the design of that application will directly impact those behaviors. Thus, a design that is faster to use may not, in fact, lead to the best security outcome. Similarly, if the most secure behaviors are time consuming, users will find less secure methods for accomplishing their goals.

Hence, what is missing from existing UI/UX methods are ways to consider the impact of the user interface and user experience design on application security outcomes. While there are myriad ways to measure how infrastructure, architecture, and code impacts the security of an application, there is not currently a way to evaluate the cybersecurity implications of a particular user interface. In this paper, we propose a method, the User Interface Security Assessment (UISA), for assisting designers and developers in identifying and considering the security implications of a particular interface.

Our research is inspired by human reliability measurement methods for safety critical systems, focusing in particular on the errors that people can make within any application [1,2]. Human error has been identified as a root cause of many security incidents [3]. Therefore, our method focuses on all the ways that users may not behave as expected, both from unintentional mistakes as well as intentionally avoiding unusable interactions. By identifying these errors, their implications and triggers, application stakeholders can evaluate the security risks of a particular design or application, prioritize aspects of a design that may need improvement, and consider different design ideas based on the potential impact on security. While the method could apply to any application, we are focusing in particular on enterprise applications, used within an organization by employees. These applications can have significant impact on the security of the organization and its customers, and organizations may need to choose which application to use, or customize and develop their own applications.

2. Background

Two main causes of security breaches in organizations are technical vulnerabilities and human errors [3]. While technical vulnerabilities receive a significant level of preventative attention from organizations, human errors are often overlooked. The usable security community has long argued that the usability of an application has an important impact on the security of that application. Thus, designers and developers need methods to help assess usable security issues.

2.1 Human Reliability Methods

Human Reliability Assessment methods are a well-established research area within safety-critical systems. Numerous methods have been proposed for organizations to identify and measure the impact of human errors from several different perspectives including process, psychology, response time, task and socio-technical influence [4]. HRA methods follow several different approaches to identifying and measuring human error. Some methods rely on prior measured data of the types and prevalence of particular

USENIX Symposium on Usable Privacy and Security (SOUPS) 2020. August 9 -- 11, 2020, Boston, MA, USA.

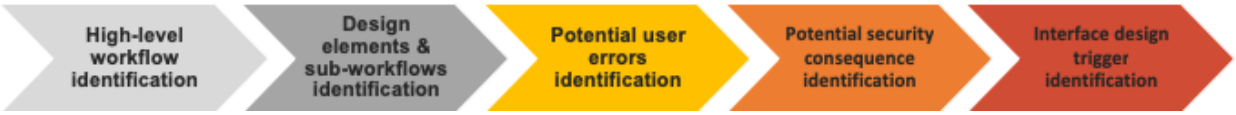

Figure 1. Overview of the UISA Process

types of human errors to calculate specific risks in a particular context. Others rely on expert evaluations of the rankings and likelihood of those errors. HRA methods are further characterized based on their level of attention to contextual factors, such as environment conditions, stress and time pressure. The greater the focus on contextual factors, the more accurate the method, yet also the more complex.

Since 2013, multiple researchers have suggested development of a systematic approach for prediction of data breaches caused by human error. [2,3,4]. As one of the first efforts in this area, Gu et al, 2014, suggested use of the Technique for Human Error Rate Prediction (THERP), which is a mature technique in human reliability analysis, in the process of information security risk assessment [4]. Similarly, Evans et al. suggested adapting HEARTS, another well-known HRA technique, as a systematic approach for human reliability and error assessment in the information security area [3].

Evans, for example, proposed IS-CHEC based on HEARTS, involving identifying error producing conditions (EPCs) for security incidents reported in an organization [5]. These EPCs include situations such as users' unfamiliarity with infrequent situations where an important action is required, and little or no independent checking or testing of output. Documenting these potential human-related causes for security incidents will allow organizations to identify common causes and potentially address them.

While these two proposed techniques do focus on human errors, they do so at a very high level. For example, the methods outline several causes of human error, such as a mismatch between an operator's model of the world and that imagined by a designer, requirement for a user to learn new skills or procedures, or shortage of time for error control. Thus, while these methods may identify that an interface design is a factor in human error, they do not help designers and developers assess or target particular design or interface decisions and identify potential areas for improvement. We aim to fill this gap with our method. Additionally, while our method is inspired by existing HRA methods, the method procedures and outcomes are specific to the behaviors people have when interacting with an interface, and focuses attention on those interactions with security implications.

2.2 Usable Security Methods

Within the usable security community a number of researchers have proposed methods for helping people better design and assess both usable and secure software. For example, at the early design stage, Cranor's Human in the Loop framework (HITL) helps guide designers to consider how and where humans should be involved in security-related decisions, to establish the interaction and interface requirements for an application [6]. Faily and Flechais suggest exploration of usable security issues of interface design through the use of misusability scenarios for design evaluation [7]. They suggest that focus on security misuse could prompt designers to think about the various ways that users could behave in unexpected or undesirable ways.

Many studies of various systems within the usable security research community have produced general design guidelines for improving similar systems. For example, a common guideline is to have safe and secure defaults [8]. Guidelines have further led to the development of specific heuristics for use in heuristic evaluations, such as for IT security management tools [9]. These kinds of guidelines and heuristics are a useful source for trained designers to consider the security impact of their design decisions. However, they may be too high level and abstract to support concrete decisions of designers, and few UX designers and developers have usable security knowledge to even be aware of them.

The most related method to our effort is a software quality attributes model by Gordieiev et al. [10]. They proposed a usable security evaluation method aimed at ensuring that a required level of security and usability is maintained, while maximizing both. Their method is attribute-based, relying on evaluators to assess a set of usability attributes of an application, such as recognizability and learnability, as well as a set of security characteristics, such as confidentiality, integrity, and non-repudiation based on a limited set of questions regarding the application. While the goal of the method is to assess usable security specifically, this attribute approach evaluates an entire application and would not help stakeholders understand the impact of particular design decisions or sets of decisions. The method also does not take into account human error or non-compliance.

3. User Interface Security Assessment

Our method was motivated from our own experiences and research results in usable security. While designers may be well trained in methods for considering and improving the user experience, they are rarely experienced in considering the security implications of those design choices. On the other hand, security engineers are likely to overlook the impact of human behavior within an application, and have

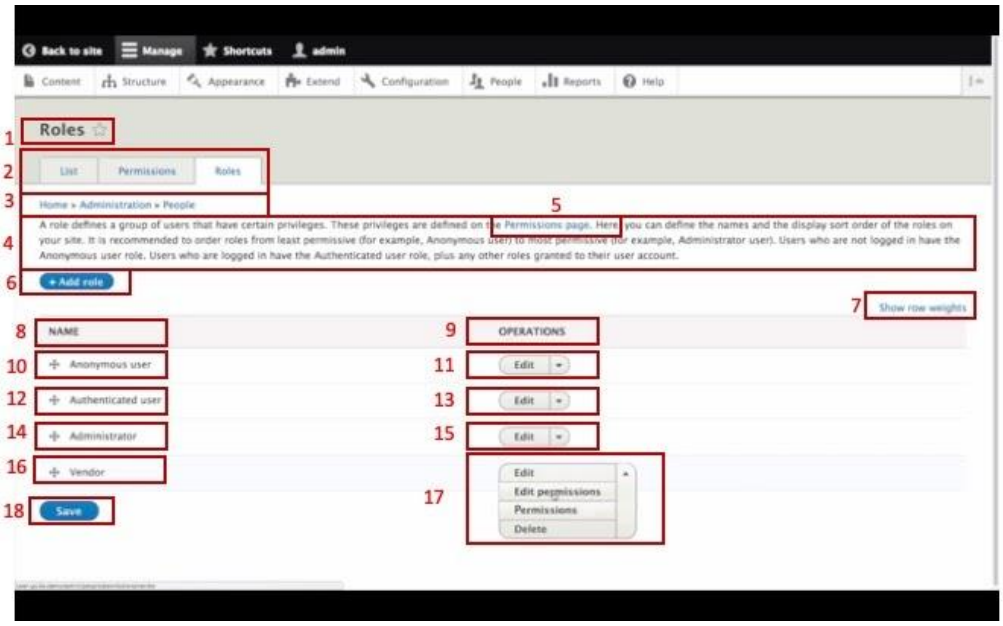

Figure 2. Sample screenshot from an example webpage content management system.

unrealistic expectations as to what users are able or willing to do. Software developers may not be familiar with either usability or security engineering, and know how to consider both of those issues together. Thus, our aim is to provide sufficient assistance based on these varied backgrounds to help a variety of stakeholders in assessing the security implications of design, as well as provide a way to structure conversations between stakeholders with different concerns regarding usability and security.

The method also focuses specifically on identifying the implications of interface and interaction design decisions, and in particular, the ways in which users will behave in imperfect or unexpected ways. Our aim is to provide sufficient assistance to identify the errors that can occur within user interface designs, consider their security implications, and discuss the potential triggers for those errors based on common design guidelines.

As illustrated in Figure 1, the User Interface Security Assessment (UISA) method has 5 steps, which we discuss in detail below. Similar to many HRA methods, each step of the method is supported by a handbook to structure the assessment, and provide guidance to the evaluators with different levels of UI/UX design background or security knowledge who may perform the assessment. We will provide examples throughout the below method of that handbook, which we expect to refine based upon community feedback, studies of use, and experience. We now describe each step of the process, using an example to illustrate the actions and expected outcomes of the assessment.

1. ***Identify target workflows and particular tasks and screens of an application***

The first step of the process is to identify the particular workflows or screens of the application as the scope of analysis. While most applications support a variety of tasks and activities, with a number of screens, dialogs, and interactions involved, an evaluation process requires a breakdown of the application to smaller components. To narrow the scope of the evaluation, the first step is to focus on a key set of tasks for the assessment. Breaking down an application to tasks and workflows increases the feasibility of the evaluation process by decreasing the process complexity. The evaluation process can be repeated multiple times to cover all the tasks and workflows in an application or only focus on a specific workflow, such as when an application is being modified or expanded.

To illustrate our method in this paper, we will refer to the simple example in Figure 2, which is a screenshot for a website content management system, such as Drupal. We created several such screenshots from Drupal's online tutorials, and slightly modified the interface design to make it a more suitable example for illustration of our method. The screenshot shows one of two screens from a workflow for changing the role permissions of various user roles.

While in this simple example, the workflow consists of a single user and two screens, more complicated workflows could include more screens and even multiple users. For example, similar systems could involve users requesting certain permissions, and workflows for administrators or managers to approve them. While evaluators can focus the assessment on whatever part of the workflow they choose, it

will be important that evaluators understand the outcome of a user's actions on the system and the information contained within it, as well as the implications for downstream users.

2. Elaborate all of the user interface elements of target workflows

The next step of UISA is to identify all of the interface components involved in the chosen workflows. This includes elaboration of every element where users provide input, navigate or perform activities. For example, in Figure 2, we outline all of these elements in red, including navigation links and tabs, drop down menus, and buttons for the role modification screen. For the handbook, we developed a general list of these elements based upon a comparison of design component lists on UI design websites and an observation of common elements in interface designs. Assessors should create a list of each of these elements to support the next stage of the evaluation. Table 1 lists a subset of the elements in this example. Future tool support for this method could either automatically determine this list from the application code, or allow evaluators to choose from a list of UI elements.

Design Element	Instance	Potential User Errors
Drop Down	11,13,15,17 (Identical)	Do not make a choice
		Choose wrong option
Link	5	Not pressed when should be pressed
		Pressed when should not be pressed
		Pressed multiple times
Button	6	Not pressed when should be pressed
		Pressed when should not be pressed
		Pressed multiple times

Table 1: Potential errors for some of the identified design elements of Figure 2 identified based on the UISA guideline list of Design Elements

3. Identify the potential errors

After all the design elements for the target screen/workflow have been identified, the next step is to explore all the potential errors a user could make. The error identification stage has two substeps. First, listing all the potential errors users can make while interacting with each single element of the workflow; second, considering the combination of elements in the workflow and listing all the errors users could make interacting with the combination.

In the UISA method, we emphasize that users can make many kinds of mistakes, from unintentional slips and typos, to making the wrong choices due to misunderstandings, to higher level decisions to find ways of accomplishing tasks outside of the intended workflow. UISA emphasizes that no possible error should be overlooked during the assessment process, so that the short and long term security impacts of any possible error be considered. The motivation for this emphasis is our observation that in an organizational context, developers may believe that users are not going to make obvious errors nor make the type of errors covered in organizational guidelines or training materials. Thus, our method first requires evaluators to identify all possible errors, and then examine the implications of those errors, before filtering any out.

To support evaluators in identifying possible errors, as a part of the UISA handbook, we developed a list of possible user errors for each type of interface element, and a list of users' common workflow-level mistakes. For example, as listed in the Table 1, element level errors are conditions such as when users click the wrong check box, menu option, or button. Users could also neglect to provide input, or provide the wrong input. At a higher level, as examples of workflow level errors, users could abandon the task before completing it, or find an alternative (and possibly less secure) way to accomplish a task.

In the context of our example of modifying role permissions, one of the potential design element errors is that the user checks a checkbox and grants a role unintended permissions, such as the ability to view the files overview page. This error could give too much access to the contents of the website to unintended users.

At the workflow level, one of the possible errors is that the user performs an unintended task instead of the original task. In Figure 2, this workflow error would occur as a result of a design element error due to the similarly labeled, but different, edit permission menu options that take the user to either role permissions or site permissions (item 17 in Figure 2).

4. Identify the security consequences

After identifying the design components and possible user errors, evaluators will then consider the security implications of all of these potential errors. To guide evaluators in thinking through these implications, we created a set of questions to ask for both element-level errors and workflow-level errors. If the answer to any of these questions is positive, then evaluators should describe the security-related implication in their own words, to make the implications concrete for the particular application.

For example, in Table 2, following the previous example of the workflow error with the permissions page navigation, as the evaluator went through the questions, they would answer yes to "Could process/workflow fail because user performs a wrong action or cannot provide the required input". Following the process, the evaluator would describe the implica-

tions of this, such as in Table 3, particularly that it could cause the user to alter the wrong permissions and grant inappropriate access to modify or view content on the website.

Potential Security Consequence	Consequence Type
Could wrong next level decisions be made due to inaccurate or incomplete information?	Integrity
May next level decisions not be possible due to lack of information?	Integrity and Accessibility
Could the process/workflow fail because the user performs a wrong action or cannot provide the required input?	Accessibility
Could the user adopt a non-legitimate alternative solution because of the interface complexity, too much restriction or timely process	Confidentiality
May data integrity get damaged due to wrong input or wrong actions leading to modification of data or wrong input?	Integrity
Could incorrect actions or processes result such as revoking access, data transfer, etc. happen?	Integrity and Accessibility and Confidentiality

Table 2: Examples of questions to guide decisions regarding high-level potential security consequences

Instance	Potential Errors	Potential Security Consequences
17	Do not make a choice	-Incorrect access to content granted when permissions are not modified as desired -Insufficient actions may provide a possibility for damage to content (data) integrity
	Choose wrong option	-Failure to navigate to the correct page to edit intended permissions -Wrong permissions may be chosen, resulting in unintended access. -Improper changes may provide a possibility for damage to content (data) integrity

Table 3: List of some element-level potential security consequences in the context of our example

If evaluators determine that errors do not have security implications, these errors can now be filtered out and not considered further. While they may impact the use of the application, they will not contribute to risks related to security.

5. Identify the triggers

The final stage of the UISA process is identification of the underlying design choices which may increase the likelihood of the identified errors with security consequences. Users can make slips such as a typo in even the most well-designed textbox. Yet, users are more likely to intentionally or unintentionally provide the wrong input if the labels are poor or formatting of the text is unclear or not checked by the application.

To reduce the overhead of identifying triggers, and help provide focus to those less familiar with interface design, UISA provides a set of triggers derived from common interface design guidelines, as well as usable security guidelines.

Instance	Potential Errors	Design Triggers for Element Level User Errors
11	Do not make a choice	Relevant option exists but worded unclearly
	Choose wrong option	Irrelevant option is listed as a choice
		Relevant option exists but worded unclearly so user misinterprets

Table 4: List of some element-level triggers for two user errors leading to security consequences in our example interface

Back to our example interface from Figure 2, the design trigger for this error is due to menu labeling. It is unclear which option should be used to change the role permissions, as opposed to the site permissions.

After determining the triggers, assessors should prioritize the triggers based on the severity and likelihood of impact in order to identify which design changes are the most important and should be addressed first. For example, the navigation confusion in the drop-down menu in Figure 2 could be addressed by changing labels to be more descriptive, such as “Edit Role Permissions” and “Site Permissions”. Once the redesign is complete, assessors could use the method to evaluate the new design to find any new security concerns.

Outcome of the assessment

Assessment based on the UISA Guidelines identifies security risks from user errors in current or potential user interface designs, as well as guides designers in modifying design elements to eliminate or minimize the identified security risks. Each iteration of the method will result in a list of relevant workflows and design elements, potential errors, resulting security concerns, and the triggers that could cause each concern. This list can be used to guide designers in identifying which elements should be redesigned to mitigate the security concerns by decreasing the potential for user errors. Through multiple iterations of the assessment steps and completion of suggested design changes, the interface

should be optimized to minimize usage errors according to the concerns in the handbook.

4. Limitations of the method

Similar to many HRA methods, the UISA method relies on the use of a handbook to guide evaluators in determining possible errors and consequences. If this handbook is not sufficient or complete, evaluators may overlook important issues. Yet, the handbook in its current format is already lengthy and could be time consuming to use, especially for those less familiar with interface design. Our next stage of this research is to perform evaluations on example systems using the handbook to make improvements, identify what is missing, and attempt to find a balance between completeness and over-specification of errors and triggers. Elements of the handbook could also be customized for each organization to focus on the elements, implications, and triggers that are common to that organization's software.

Additionally, the repetitive stages of the method, such as comparison of each error against the whole trigger list could make the evaluation process time consuming and error prone. To address this issue we are planning to develop an assessment tool based on the UISA method to guide evaluators through the process and ease the burden of documenting the errors, implications and triggers.

5. Conclusions

We are developing the User Interface Security Assessment method, to aid designers and developers in evaluating the potential security implications of human behavior. We hope that the method can assist organizations in understanding how user interactions within enterprise applications can impact security, help stakeholders to identify specific areas where human behavior could lead to security problems, and reason about and discuss the causes and potential interface solutions. Our next steps are to evaluate this method for its usability and usefulness with evaluators with different backgrounds, and continue to iterate and improve the handbook for use in this method. We plan to perform feasibility studies of the handbook in the context of different organizations and applications to ensure the lists of elements, errors, consequences and triggers are comprehensive and address most of the common interface design-related sources of security incidents.

We also plan to build upon this method to develop a metric which will quantify the security risk associated with a particular design, similar to the metrics that are part of many Human Reliability Assessment methods. Metrics will further allow application stakeholders to prioritize the most troublesome workflows and designs, to compare designs or applications against each other, to track improvements as interfaces are redesigned, and to report overall assessment outcomes throughout an organization. We welcome review and feedback of the handbook, and suggestions for improving or expanding upon this method.

6. References

1. Bell, J., & Holroyd, J. (2009). Review of human reliability assessment methods. *Health & Safety Laboratory,* 78.
2. Evans, M., Maglaras, L. A., He, Y., & Janicke, H. (2016). Human behaviour as an aspect of cybersecurity assurance. *Security and Communication Networks*, *9*(17), 4667-4679.
3. Evans, M., He, Y., Maglaras, L., & Janicke, H. (2019). HEART-IS: A novel technique for evaluating human error-related information security incidents. *Computers & Security*, *80*, 74-89.
4. Gu, T., Li, L., Lu, M., & Li, J. (2014, August). Research on the calculation method of information security risk assessment considering human reliability. In *2014 10th International Conference on Reliability, Maintainability and Safety (ICRMS)* (pp. 457-462). IEEE.
5. Evans, M., He, Y., Luo, C., Yevseyeva, I., Janicke, H., & Maglaras, L. A. (2019). Employee perspective on information security related human error in healthcare: Proactive use of IS-CHEC in questionnaire form. *IEEE Access*, *7*, 102087-102101.
6. Cranor, L.F. (2008). A framework for reasoning about the human in the loop. In *Proceedings of the USENIX Workshop on Usability, Psychology and Security*, April, 2008.
7. Faily, S., & Fléchais, I. (2016). Finding and resolving security misusability with misusability cases. *Requirements Engineering*, *21*(2), 209-223.
8. Garfinkel, S. and Lipford, H.R. (2014). *Usable Security: History, Themes, and Challenges*. Morgan & Claypool.
9. Jaferian, P., Hawkey, K., Sotirakopoulos, A., Velez-Rojas, M., & Beznosov, K. (2011, July). Heuristics for evaluating IT security management tools. In *Proceedings of the Seventh Symposium on Usable Privacy and Security* (pp. 1-20).
10. Gordieiev, O., Kharchenko, V., Vereshchak, K.: Usable security versus secure usability: an assessment of attributes interaction. In: Proceedings of the 13th International Conference, ICTERI 2017, 15–18 May, Kyiv, Ukraine, pp. 727–740 (2017)