



**HAL**  
open science

# A Generic Multimodels-Based Approach for the Analysis of Usability and Security of Authentication Mechanisms

Nicolas Broders, Célia Martinie, Philippe Palanque, Marco Winckler, Kimmo Halunen

## ► To cite this version:

Nicolas Broders, Célia Martinie, Philippe Palanque, Marco Winckler, Kimmo Halunen. A Generic Multimodels-Based Approach for the Analysis of Usability and Security of Authentication Mechanisms. HCSE 2020 - 8th International Conference on Human-Centered Software Engineering - IFIP WG 13.2 International Working Conference, Nov 2020, Eindhoven/ Online, Netherlands. pp.61-83, 10.1007/978-3-030-64266-2\_4. hal-03079818

**HAL Id: hal-03079818**





**<https://hal.science/hal-03079818v1>**

Submitted on 17 Dec 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# A Generic Multimodels-Based Approach for the Analysis of Usability and Security of Authentication Mechanisms

Nicolas Broders<sup>1</sup>(✉), Célia Martinie<sup>1</sup>(✉) , Philippe Palanque<sup>1</sup> ,  
Marco Winckler<sup>3</sup> , and Kimmo Halunen<sup>2</sup> 

<sup>1</sup> ICS-IRIT, Université Toulouse III-Paul Sabatier, Toulouse, France  
{nicolas.broders, celia.martinie, philippe.palanque}@irit.fr

<sup>2</sup> VTT Technical Research Centre of Finland Ltd., Espoo, Finland  
Kimmo.Halunen@vtt.fi

<sup>3</sup> Université Côte D'Azur, Nice, France

**Abstract.** Authentication is a security function, added on top of an interactive system, whose role is to reduce organizations and users' risks to grant access to sensitive data or critical resources to unauthorized users. Such a security function interfere with users' goals and tasks by adding articulatory activities, which affect each dimension of usability. In order to mitigate their negative effect on usability, security functions must be designed following a User Centered Approach. In order to ensure their efficiency in terms of security, security processes have to be followed. With this respect, this paper focuses on the representation of user tasks (using task modelling techniques) to be performed during authentication. For security aspects, we propose the use of an approach called "attack trees" which represents threats and their effect. To integrate both aspects in a single framework, we propose an extended task modelling technique that is able to represent explicitly security threats and their potential effect together with users' tasks performed during authentication. We show how such models can be used to compare the usability and the security of different authentication mechanisms and to make explicit conflicts between these properties. We exemplify the use of the approach on two sophisticated authentication mechanisms demonstrating its applicability and its usefulness for representing and assessing in a single framework, usability and security of these security mechanism.

**Keywords:** Usability · Security · Tasks descriptions · Authentication

## 1 Introduction

Authentication is one of the most common and repetitive activity users have to carry out every day when accessing information technologies. Security mechanism are designed to protect users' assets by preventing a straightforward access to them, thus adding complexity to the user interaction with the system and ultimately degrading the system usability and, in particular the users' performance [48]. For example, early studies have

demonstrated [21] that users have an average of 25 accounts requiring identification while user's capability of recalling multiple passwords is still questioned [7]. In their early work, Adams and Sasse [1] advocated that "Security Needs User-Centered Design". The User Centered Design (UCD) process, as defined in ISO 9241 part 210 [30], determines that the identification of the users' goals and needs is one of the key elements to produce usable interactive system. The same ISO 9241 standard in the part 11 [29], defines usability by three dimensions: efficiency, satisfaction, and effectiveness. Efficiency is assessable via user testing for instance counting the time needed to complete tasks and the number of user errors occurring during the performance of these tasks. User satisfaction can be assessed by direct observation of users (ex. positive/negative comments and analysis of facial expression) and/or using specialized questionnaires such as the System Usability Scale (SUS) [11]. For the effectiveness, the best way to assess it according to [19] is to identify exhaustively the tasks the users have to perform in order to reach their goals and to check that each task is supported by the system [34]. Assessing the cost of user errors and the cost of recovering from them is also an important aspect that can be addressed using dedicated structuring [44] or adding specific extensions to task models [17].

Even though a lot of work has been carried out on evaluating the usability of authentication mechanisms two by two [12] or more globally [8], the current state of the art demonstrates that the design and development of security mechanisms does not systematically takes into account the three before mentioned dimensions of usability, and in particular the effectiveness. Indeed, one of most cited analysis from Bonneau et al. [8], which compares the usability of 35 different authentication mechanisms, does not recognize effectiveness among the 8 different aspects covering by the so-called « Usability Benefits». It is also the case in more recent work that classifies properties that have to be supported by user authentication methods and which does not contain effectiveness [26].

This paper demonstrates how tasks models can be used to systematically represent the quantity and the complexity of work that users have to perform to complete an authentication on a given system. Moreover, we propose to extend a task-modelling notation to represent explicitly the security threats and their effect. The proposed use of this integrated representation of security and usability aspects consists in modelling users' activity for authentication on different authentication mechanisms. These models can then be used to compare the usability and security of the mechanisms under consideration both in terms of security and usability. As the design of these systems usually consists in a trade-off between these two properties, the proposed approach provides means to make these trade-off explicit. This combined representation makes it also possible to identify potential countermeasure to mitigate the effect of threats but also to identify ways to improve the usability of the mechanisms without degrading their security. This aspect related to design of usable and secure authentication mechanisms is beyond the scope of this paper that focusses on the analysis part of extant systems.

The paper is structured as follows: Sect. 2 presents the related work on methods for the analysis of usability and security as well as their main limitations in terms of identifying exhaustively user tasks and potential threats. Section 3 presents an extension to the task modelling notation called HAMSTERS-XL which was specifically designed to represent

the possible threats and their effects for each relevant task in the task model. Then, using a simple case study of password authentication, we illustrate how the combined description of tasks and threats enable to identify which threats are covered or not covered by the authentication mechanism. Section 4 presents the applications of these notations to the case study of comparing the usability and security of two authentication mechanisms. Section 5 concludes on the main benefits of the proposed approach.

## 2 Existing Approaches for the Analysis of Usability and Security of Interactive Systems

Research and industry have been producing, at a very high pace, diverse authentication mechanisms with increasing complexity for the users. The current practice in authentication is promoting multi-factor authentication methods (i.e. the access to the system is granted after the user provides two or more pieces of evidence, the so-called factors, to the authentication system). Even though the adoption and acceptance by users is questioned [45], other user studies (based on Mechanical Turk users) suggest that two factors authentication mechanisms are, overall, perceived as usable [16]. Most of the work concerning the usability of security mechanisms lies in one of the following categories:

- analysis of user behavior when confronted to security mechanisms (e.g. user behavior when password expires [25]),
- recommendations to enhance usability of a specific existing security mechanism (e.g. authentication in a smart home [28]),
- comparative studies of the usability of security mechanisms (e.g. which authentication mechanism would better replace CAPTCHAs [23]),
- techniques that might improve users' performances (e.g. improving users' memorability of authentication mechanisms [39]).

As we shall see, most of the existing work focuses on specific security mechanisms (such as authentication) on a case-by-case basis and they mainly address user performance and/or user satisfaction. Nonetheless, the analysis of effectiveness requires the identification of all possible user tasks [19]. Hereafter, we discuss a few generic methods that can be used to analyze the tradeoffs between usability and security of interactive systems. Then, we highlight the methods explicitly addressing the effectiveness.

### 2.1 Generic Methods for the Systematic Analysis of Usability and Security

Despite of a large literature, very few work propose generic methods that can be used to compare diverse types of security mechanisms and systematically assess the trade-offs between security and usability. Alshamari [3] highlights the recurrent conflicts between usability and security. That author proposes a generic process to identify these conflicts between usability and security at design time, and to select a strategy to handle them using a decision support system for eliciting requirements. Other works employ inspection methods to analyze the effect of security mechanisms on the usability of interactive

system. Braz et al. [10] propose a set of heuristics and an inspection method for the analytical evaluation of the effects of security mechanisms on the system's usability. Alarifi et al. [2] propose a structured inspection model dedicated to the analysis of usability and security of e-banking platforms. Bonneau et al. [8] propose a set of heuristics for comparing usability and security benefits between several authentication techniques. Such inspection methods and heuristics provide support to compare security mechanisms and to make tradeoffs during design. However, these approaches cannot ensure an exhaustive coverage of all possible user actions with the system and they might fail in detecting problems related to specific scenarios. Ben-Asher et al. [6] propose an experimental environment to collect systematically any possible user behavior facing security mechanisms. They propose to use the output of user tests run in the experimental environment to explore possible tradeoffs between security and usability for the system under design. That approach can help to identify usability problems and/issues with security mechanism with tasks performed by users during the experimental phase. As such, the logistic required to run the user test limits the coverage of the study to a small subset of tasks that are possible with the system.

## **2.2 Generic Methods that Take into Account Effectiveness**

All the existing approaches for systematically assessing usability and security rely on scenarios. A "scenario describes a sequence of actions and events that lead to an outcome. These actions and events are related in a usage context" [47]. Kainda et al. [32] propose a security-usability analysis process. They first present a so called "security-usability threat model" which is a set of important factors for the analysis of usability and security. Then, they produce both threat and usage scenarios that are used to assess whether (or not) the set of elicited factors are met by the security mechanisms. Faily and Fléchais [18] propose a scenario-based approach for assessing the effect of security policies on usability. The specificity of this approach is that it is based on "misusability" cases which are produced using systematic questioning of system development and deployment documentation (e.g. architecture specification, user manual...). This approach aims to inform re-design of security mechanisms.

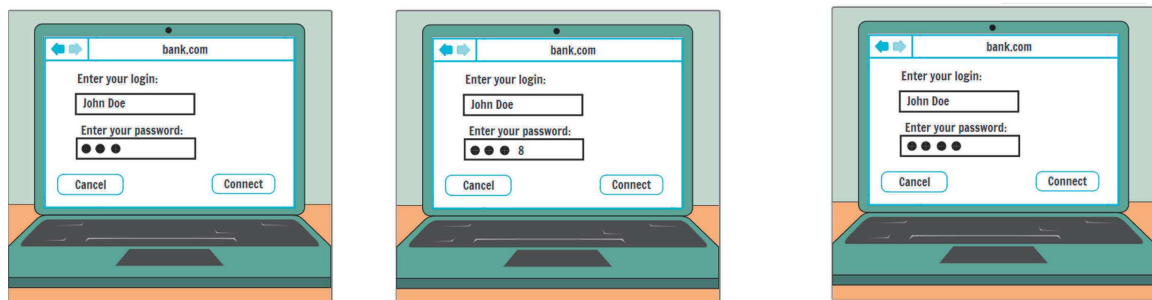
Like other inspection methods, scenario-based methods allow to identify security threats and support the analysis of the range of user behaviors when facing threats. Scenarios bring the attention to specific tasks, thus promoting systematic inspections and reducing the risk of finding problems only by change. However, because scenarios are focused on specific tasks they do not cover all possible temporally ordered sets of user tasks and, as a consequence, the analysis of the effect of potential target threat on user tasks may be incomplete.

## **3 Describing Security Threats and Their Effects with Task Models**

This section introduces an illustrative authentication mechanism and its corresponding description using a task model notation called HAMSTERS-XL. As we shall see, the task model is enriched to represent threats and the effects of these threats.

### 3.1 Illustrative Example: Text-Based Login with Short Display of User Input

Figure 1 presents a storyboard of some of the users actions required for logging in an online bank to check the accounts' balance. The middle figure shows that user input is shown for 2 s, thus providing feedback and potentially preventing user mistakes. After the 2 s, a dot visually conceals the manual entry to ensure security (right-hand side Figure of Fig. 1).



**Fig. 1.** Storyboard of a text-based authentication featuring short display (2s) of user entries.

### 3.2 Modeling Tasks

The systematic identification of potential security threats on user actions and the effects of authentication mechanisms on user tasks requires the description of:

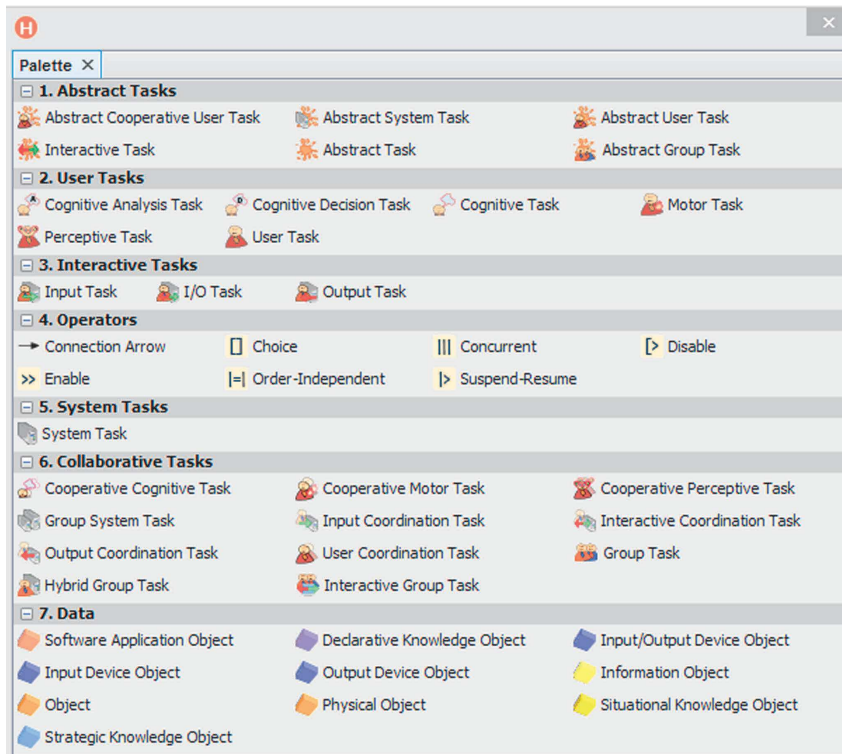
- user actions: a threat can arise from a type of user action, e.g. drawing a gesture password on a tactile screen is subject to smudge attacks whereas typing a password on a keyboard is subject to keylogging attack,
- their temporal ordering and/or temporal constraints: a threat can arise from the specific ordering of user actions, e.g. the user takes too long to input a password,
- the information, knowledge and objects being manipulated during these actions: a threat can arise from an information, knowledge or object that the user has lost, forgotten or misused, e.g. a credit card lost in a public space.

Hereafter, we describe how these elements are taken into account using the task model notation named HAMSTERS.

#### Description of the Task Modelling Notation

HAMSTERS (Human – centered Assessment and Modelling to Support Task Engineering for Resilient Systems) is a tool-supported task modelling notation for representing human activities in a hierarchical and temporally ordered way [37]. The HAMSTERS notation provides support for representing a task model, which is a tree of nodes that can be tasks or temporal operators (Fig. 3 presents a HAMSTERS task models describing the users actions to login). The top node represents the main goal of the user, and lower levels represent sub-goals, tasks and actions. Task types are elements of notation that enable to refine and represent the nature of the task as well as whether it is the user or





**Fig. 2.** Palette of the elements of notation of HAMSTERS

the system who performs the task. The main task types are abstract, user, interactive and system tasks (parts numbered 1, 2, 3, 5 in Fig. 2).

Abstract tasks (part numbered 1 in Fig. 2) provide support to describe sub-goals in the task model. They also provide support to describe tasks for which the refinement is not yet identified, at the beginning of the analysis process. User tasks (part numbered 2 in Fig. 2) provide support to describe the detailed human aspects of the user activities. User task types can be refined into perceptive, motor, cognitive analysis, and cognitive decision tasks. For example, the user may perform a motor task (such as grabbing a card) or cognitive task (such as remembering a PIN code). Such refinement enables the analysis of several aspects of the tasks performed by the user such as cognitive load, motor load, required perceptive capabilities. Such refinement also enables to identify possible threats that can be associated to specific types of user actions. Temporal operators are used to represent temporal relationships between sub-goals and between activities. Interactive tasks (part numbered 3 in Fig. 2) provide support to describe tasks that are action performed by the user to input information to the system (interactive input task) or action perform by the system to provide information to the user and that are meant to be perceived by the user (interactive output task). Interactive input/output tasks provide supports to describe both cases. System tasks (part numbered 5 in Fig. 2) provide support to describe the tasks that the system executes. The system may execute an input task, i.e. the production and processing of an event produced by an action performed by the user on an input device. It may also execute and output task, i.e. a rendering on an output device (such as displaying a new frame on a screen). The system may execute a processing task (such as checking the user login and password).

In addition to elements of notation for representing user activities and their temporal ordering, HAMSTERS provides support to represent data (e.g. information such as perceived amount of money on an account, knowledge such as a known password), objects (e.g. physical objects such as a credit card, software objects such as an entered password) and devices (e.g. input devices such as keyboard, output device such as a screen) that are required to accomplish these activities (part numbered 7 in Fig. 2).

HAMSTERS and its eponym interactive modelling environment is the only environment providing structuring mechanisms as real-life models are usually large and reuse is useful [36]. HAMSTERS also provides elements of notation to identify and describe human errors [17], which is useful to detect possible user errors and then re-design the system or estimate the costs of recovering the error. Furthermore, HAMSTERS provides functions to extend tasks types and data types if required for the analysis [37]. For all of these reasons, expressiveness of the notation and modelling support, we chose HAMSTERS to be used with our approach.

### The Task Model of the Illustrative Example

Figure 3 shows the task model featuring temporally ordered user actions required for logging in as set in Sect. 3.1. First, the computer perform the task “Display the login page” (an output system task) so that the user can “See the login page” (a perceptual user task). In the sequence the user must “Fill in the username” (an abstract task, not detailed here) and “Fill in the password (an abstract task). To reach this sub-goal, the user must “Recall password” (a cognitive user task connected to a declarative knowledge object “known password”). This task is linked to the information “recalled password” (an information object) to indicate that the user was able to remembered the password.

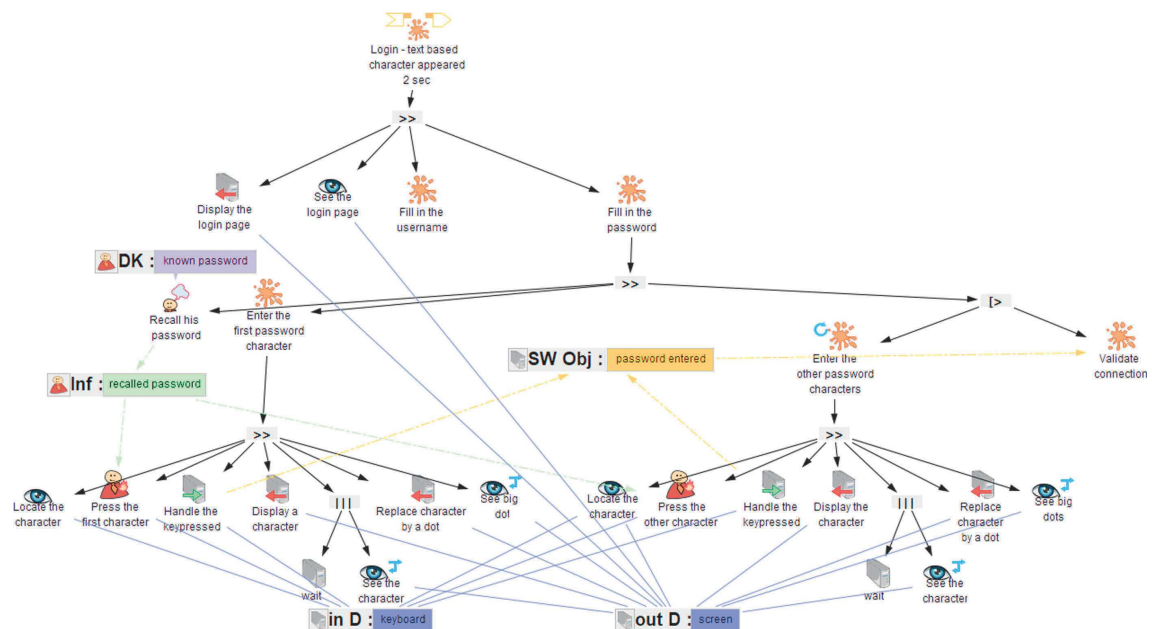


Fig. 3. Tasks model describing user actions to login

With the password in the mind, the user can “Locate the character” (a perceptive task) and “Press the first character” of the password (a motor task). This user action on



the keyboard is transmitted to a computer that will “Handle the keypressed” (an input system task), “Display a character” on the screen (an output system task) and “wait” (a system task). The waiting time of 2 s prompts the user to “See the character” (an optional perceptual user task) and detect any possible mistakes while typing. After 2 s the system will “Replace character by a dot” (an output system task) and from there the user will “See big dot” (an optional perceptual user task) instead of characters. The next task “Press the other password characters” (an iterative task) repeats as many times as characters left to complete the password. To complete the login, users must “Validate connection” (an abstract task, not detailed).

For the following figures, we will not describe again the details of the task types. You can find these task types in Sect. 3.2 describing the task modeling notation.

### 3.3 Modelling Threats and Effects

The systematic identification and representation of threats and effects of threats require a description of user actions and all possible threats that are not directly related to the user actions (e.g. network, electronic components...). Such information is essential to design and implement mechanisms to avoid or to mitigate the threats [41].

#### The Attack Tree Notation

We selected Attack Tree to address security aspects as they are major tools in analyzing security of a system [41]. They can be decorated with expert knowledge, historical data and estimation of critical parameters as demonstrated in [22] even though early versions of them were lacking formal semantics [41]. Attack tree notation is a formal method to describe the possible threats or combination of threats to a system. B. Schneier [49] provided a first description of an attack tree, where the main goal of the attack is represented by the top root node and where the combinations of leaves represent different way to achieve that goal. In the original notation, OR nodes refer to alternatives of attacks to achieve the attack whilst AND nodes refer to combination of attacks. Nishihara et al. [41] proposed to extend the notation with potential effect of attacks and with a logical operator, SAND to represent constraints in the temporal ordering of combined attacks. Other elements of the notation include a rectangle to represent an event (such a threat or attack), an ellipse to represent an effect and a triangle to represent the fact that a node is not refined. All elements of the attack tree notation are shown at Fig. 4.



Fig. 4. Elements of notation for the attack trees

#### The Attack Tree of the Illustrative Example

The Fig. 5 shows an example of attack tree that has been produced using taxonomies of cyber security threats [33] and by grouping relevant threats into categories. For example, one category of possible threats to login is eavesdropping which gathers:

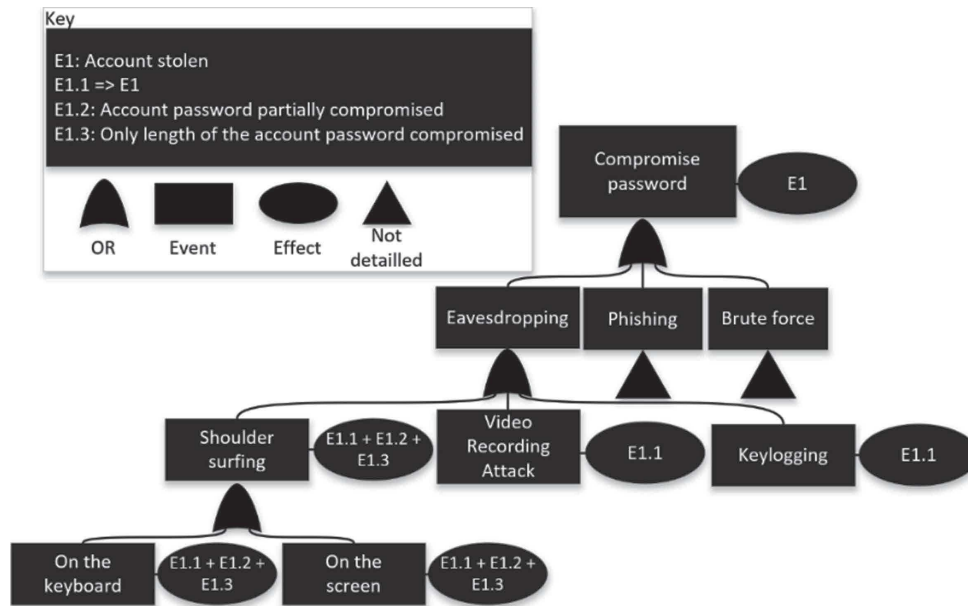


Fig. 5. Attack tree of the keyboard login authentication mechanism

- keylogging: “The key logger makes the log file of the keys pressed by the user and then sends to that log file to the attacker’s email address” [46],
- video recording attack: “The attacker uses a camera (ex. mobile phones or miniature camera) to record and analyze users typing password” [46],
- shoulder surfing: “Alternative name of spying in which the attacker spies the user’s movements to obtain the password..., how the user enters the password i.e. what keys of keyboard the user has pressed [46].

In Fig. 5, we read that an “Eavesdropping” (an event depicted as a rectangle) can be triggered by an attacker using one of the following attacks (connected by an OR node): “Shoulder surfing”, “Video recording attack”, or “Keylogging” the machine. The attacker can also “Brute force” the password or use some “Phishing” techniques but these threats are not refined in this extract (see nodes depicted by triangles).

Depending on the attacker’s effectiveness and on the user device on which the attacker will be shoulder surfing, the outcome of the attack may generate different combinations of effects (represented in Fig. 5 by the ellipse containing the effect “E1.1 + E1.2 + E1.3” respectively designing “Account stolen”, “Account password partially compromised” and “Only length of the account password compromised”).



Fig. 6. Representation of a threat (a), and effect (b), with extended HAMSTERS notation

Video recording attack and keylogging attack may generate one effect each, which are more critical (represented in Fig. 5 by the ellipse containing the effect “E1.1” designing

that “Account stolen”). The refinement of the shoulder surfing attack for two types of devices (keyboard or screen) highlights more precisely how attacks may take place and their effects depending on the targeted device. In our example, the attacks on both types of devices generate the same effects (“E1.1 + E1.2 + E1.3” possible effects on keyboard and screen).

### 3.4 Integrating Tasks, Threats and Effects

In order to represent explicitly the security threats and their effect on user tasks, we have extended the task model notation of HAMSTERS-XL. New elements of notation are: **threat**, **effect of a threat**, and the **relationships between tasks, threats and effects**. The icon representing a threat is show by Fig. 6.a, and the effect is represented by Fig. 6.b. Each threat is connected to an effect to show what effect could be the consequence of this threat.

#### The Integrated Model of the Illustrative Example

Figure 7 presents a HAMSTERS task model that embeds the representation of three eavesdropping potential threats (namely keylogging, shoulder surfing, and video recording attack) and their effects.

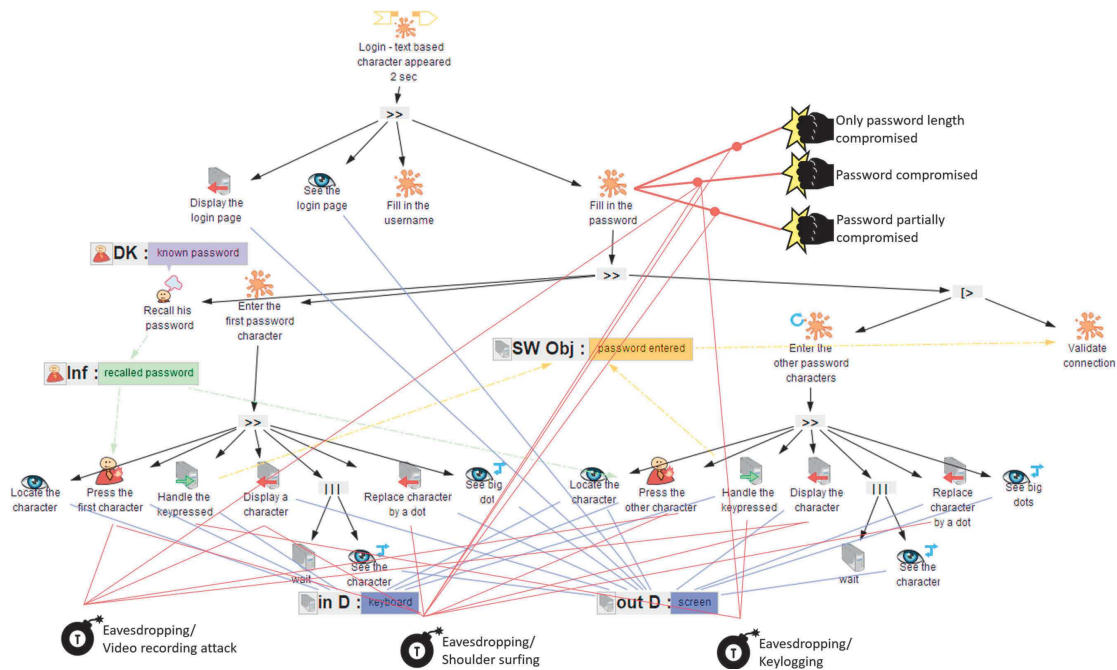


Fig. 7. Task model of the text-based login with identified threats and their effects

Each threat can be connected to one or more tasks to show that which tasks may be targets for or more threats. These relationships are depicted as strokes as we shall see in Fig. 7. Video recording attack is connected to the two user motor tasks respectively labelled “Press the first character” and “Press the other character” and the corresponding effect “password compromised”. Shoulder surfing is connected to the two motor tasks respectively labelled “Press the first character” and “Press the other character” and the

corresponding effects “Password compromised” or “Password partially compromised”, depending on the effectiveness of the hacker. Shoulder surfing is also connected to two system output tasks “Display big dot” and “Display big dots” and the effect is to have “Only password length compromised”). The keylogging attack is connected to the two input system tasks labelled “Handle the key pressed” and the effect of this threat is to have “Password compromised”.

## 4 Comparing Two Authentication Mechanisms: “Google 2 Step” and “Firefox Password Manager”

In order to demonstrate how our modelling approach can be used to compare authentication mechanisms we have designed a small case study focused on two authentication mechanisms (“Google 2 step login” and “Firefox password manager”) that are applied to an online bank application.

We first present the main user tasks (which refer to checking account balance and wire transfer) and then we present the task models including the representation of security threats for each of the two authentication mechanisms. The models are then used to support a comparative analysis to find out which mechanism has the best trade-off between usability and security. Authentication mechanisms presented in this case study (for logging in an online bank application) aim at demonstrating the approach and do not precisely reflect precisely security mechanisms deployed.

### 4.1 Presentation of the Main Tasks

Figure 8 presents a task model that describes the main users’ tasks with our online bank application: “Check accounts’ balance and make a wire transfer”.

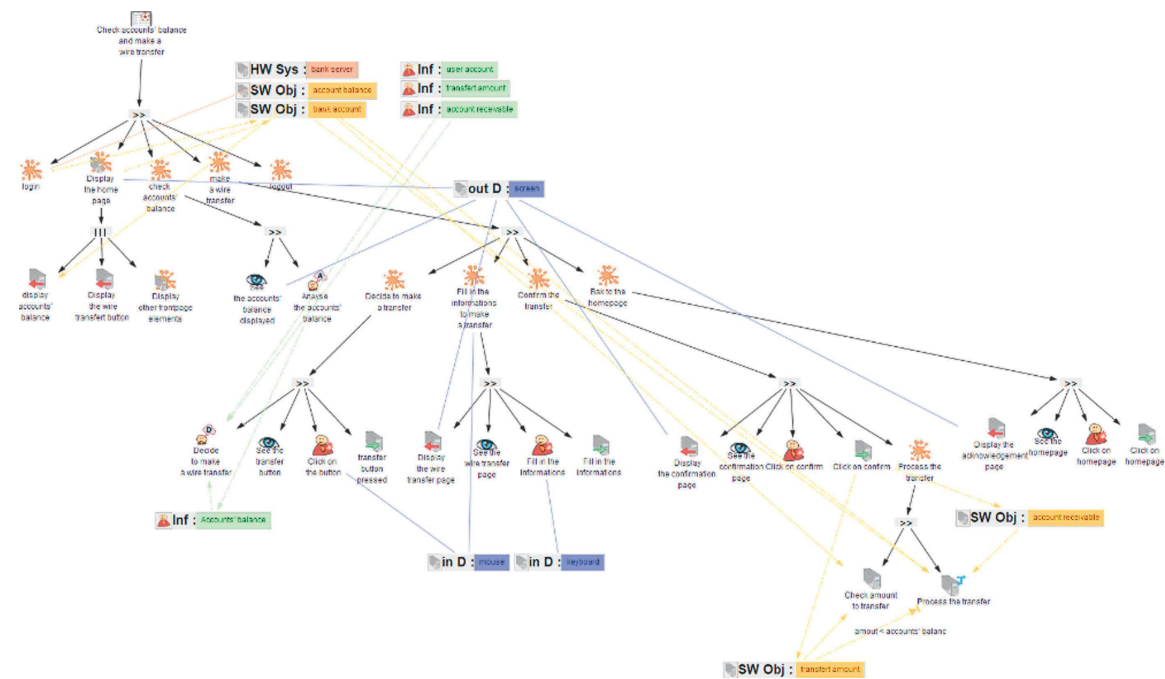


Fig. 8. Check accounts’ balance and make a wire transfer task model

- First, the user visits the web page of the bank website to perform the “login”.
- The consecutive task shows that the browser “Display the home page”. This abstract system task is decomposed into three concurrent sub-tasks: “display accounts’ balance”, “Display the wire transfer button”, and “Display other homepage element”.
- The user can then perform the abstract task “Check accounts’ balance abstract task”, which is refined in: “See the accounts’ balance displayed” and “Analyse the accounts’ balance”.
- After that, the user can perform the abstract task “Make a wire transfer”, which encompasses a sequence of four abstract tasks: “Decide to make a transfer”, “Fill in the information to make a transfer”, “Confirm the transfer” and “Go back to the home page”. The abstract task “Make a wire transfer” is refined in the four following sub-tasks: “Decide to make a wire transfer”, “See the transfer button”, “Click on the button”, and “transfer button pressed”. The abstract task “Fill in the information to make a transfer” is refined in four sub-tasks: “Fill in the information”, “Fill in the information”, and “Display the confirmation page”. The abstract task “Confirm the transfer” is refined in the following sub-tasks: “See the confirmation page”, “Click on confirm”, and “Process the transfer” which is made up of a sequence of system tasks “Check amount to transfer” and “Process” the transfer”. The abstract task “Go back to the home page” is refined in: “Display the acknowledgement”, “See the homepage perceptive task, “Click on homepage”.

## 4.2 Authentication Mechanism: Google 2-Step Verification

“Google 2-step Verification” is a multifactor authentication mechanism adding an extra layer of security on simple password-based authentication mechanisms [24]. After setting up of the mechanism, the user can connect to the account by entering a password, then Google sends a verification code to the user’s smartphone (the user needs a smartphone to complete to login), the authentication is complete when the user confirms by pressing “yes”.

### Task Model of the User Tasks with the Google 2-Step Verification

Figure 10 presents the task model for the Google 2-step verification mechanism; it contains the description of the threats and effects on user tasks that are detailed latter. The first task is to perform the text-based login. The precondition “device is not recognized” on the software object “recognized device” be true to enable users to perform the abstract task “Validate the second step”.

At the end of this step, the user receives a notification “Display the second authentication factor”. Due to space constraints, we cannot present here the refinement of the abstract task “Validate the second step” but the refinement of this sub-goal is also used to perform the analysis presented in Sect. 4.4. The main tasks of this second step are: “Make the second step necessary” for future logging, “Display the second authentication” factor, “Grab the phone” (we assume that the phone is unlocked and close to the user), “See the second authentication notification”, and finally “Press ‘yes’” to confirm the login.



## Attack Tree of the Google 2-step Verification

Figure 9 presents the attack tree of the Google 2-steps Verification mechanism. The goal of the attack is to get access to the bank account bypassing this mechanism. Three conditions are necessary (under the top level goal node, all the branches are linked by a SAND operator) to compromise the account of the user. First, the attacker needs to compromise the text-based password mechanism (left branch in Fig. 9). As in Sect. 3.3, the attacker has several possibilities to do it using eavesdropping techniques, phishing or brute force. Secondly, the attacker needs to get access to the smartphone of the user (second branch), at least temporarily, to confirm the login. There are two possibilities to achieve this sub-goal: the attacker uses eavesdropping techniques such as shoulder surfing or video recording attack and then unlocks the phone, or the attacker unlocks the phone by brute force or using a smudge attack. Finally, the attacker needs to press “yes” to confirm the login.

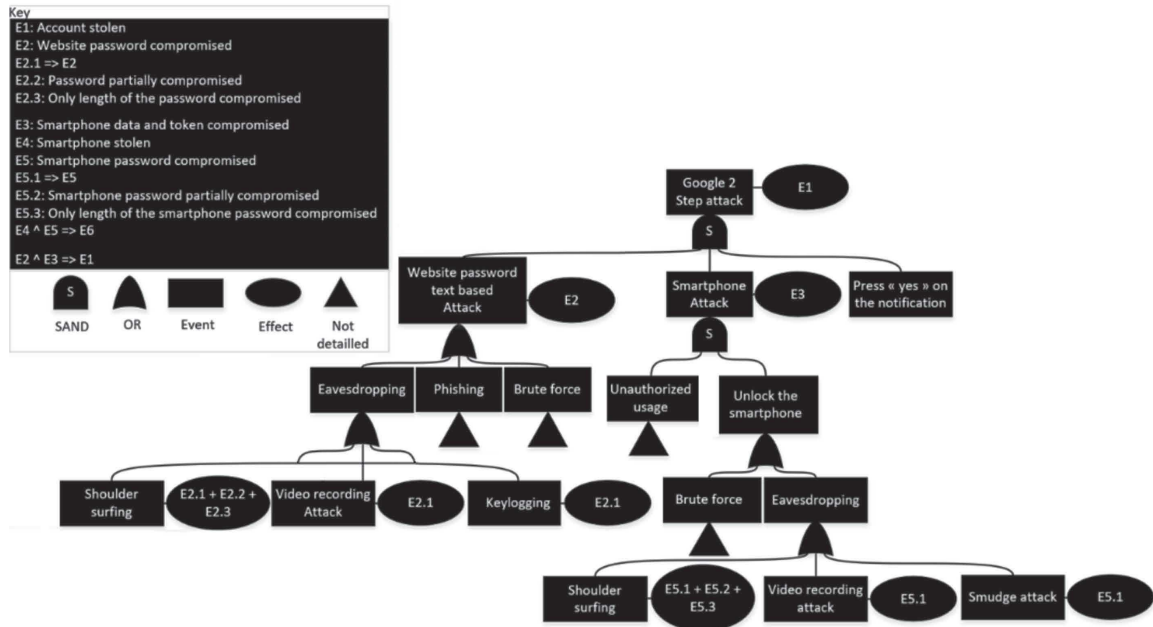


Fig. 9. Attack tree of Google 2 step mechanism

## Integrated Model of Tasks, Threats and Effects for the Google 2-Step Verification

Figure 10 describes user tasks and the possible threats and their associated effects. In this example, we show the threats and effects only on the text-based login. Those threats are of the same types than the one we presented in the illustrative example section (keylogging attack, shoulder surfing or video recording attack that may cause the length of the password compromised, password totally or partially compromised).

### 4.3 Authentication Mechanism: Firefox Password Manager

The password manager authentication mechanism is used by several internet browsers such as Firefox [20]. This mechanism allows the user to remember only one master password and the password manager securely archives and provides the relevant username



and password according to the target website for which the user has entered a username and password one time.

### Task Model of the User Tasks with the Password Manager

Figure 12 presents the task model of the user tasks, the threats and effects on user tasks with the Password Manager mechanism that are discussed latter. This mechanism involves two types of passwords: password the user needs to log in websites, and a master password to access to all passwords in the password manager. We assume that the configuration of the master password of the password manager is complete.

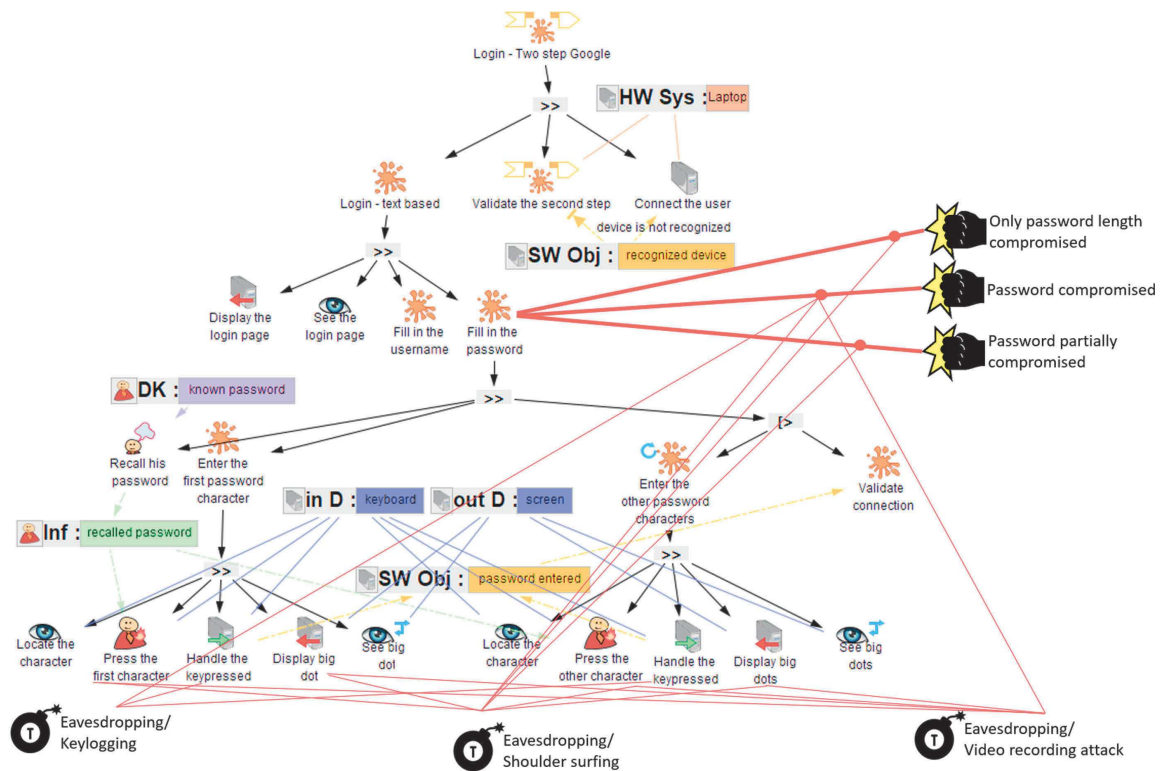


Fig. 10. Login with two step authentication task model extended with threats and effects

To login with this mechanism, there are two options (modeled using the choice operator “[ ]”) but the activation of the options depends on the value of the software object “saved password” (shown as a pre-condition on the “saved passwords” software object), as follows: for activating the first option, we assume that the password manager “does not contain website password” (a pre-condition on the software object “saved password”), so that the user must to filling in the password using the abstract task “Login – text based”. From this point, the user might “Save password” in the password manager for activating the second option, we assume that the password manager “contains website password”, so instead of filling in the website password the user perform the abstract task “Login - master password” as follows: if the user is currently logged in and in a session for less than 30 min (a pre-condition on the time duration object), the password manager detects it and the users can go to the next step; otherwise, the user has to “Enter master password”. After these login steps, the user is connected and can accomplish the tasks to complete the main goal “making a wire transfer”.

The refinement of the optional abstract tasks “save password” is presented in Fig. 13. First, the system performs the output system task “Display save password into password manager” that prompts the user to record the password; then the user can perform the tasks “See the save popup” and “Click on save”, thus enabling the system to handle the task “Click on save”. The user might perform the task “Enter master password” by following the tasks: “Display master password edit box” and “See the password master edit box”. Assuming the user recalls the password (see an incoming arrow from the “master password” declarative knowledge and “Recall master password” cognitive task), an information containing the password is produced to connecting to the system as if the user had filled in the password. After the confirmation, the system performs the task “Create session for 30 min”. The session ends up if the user closes the web browser. After that, the password manager performs the task “Crypt password in database”.

### Attack Tree of the Password Manager

Figure 11 presents the attack tree where the goal is to get access to the bank account by bypassing the Password Manager mechanism. To reach that goal, two conditions are necessary: the attacker needs to compromise the website password as well as the master password. The possible attack techniques are eavesdropping, brute force or phishing.

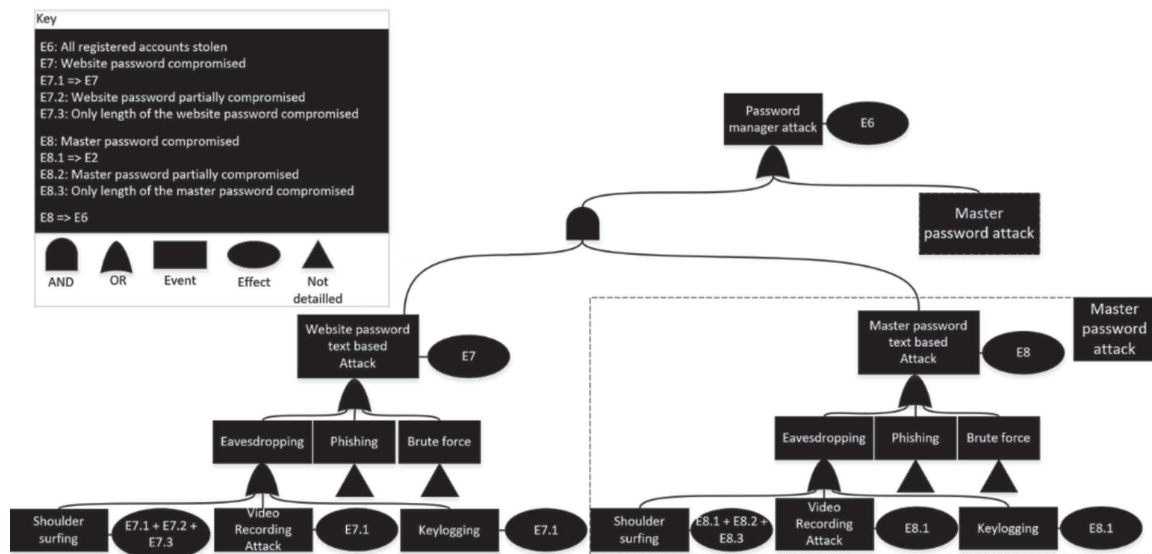


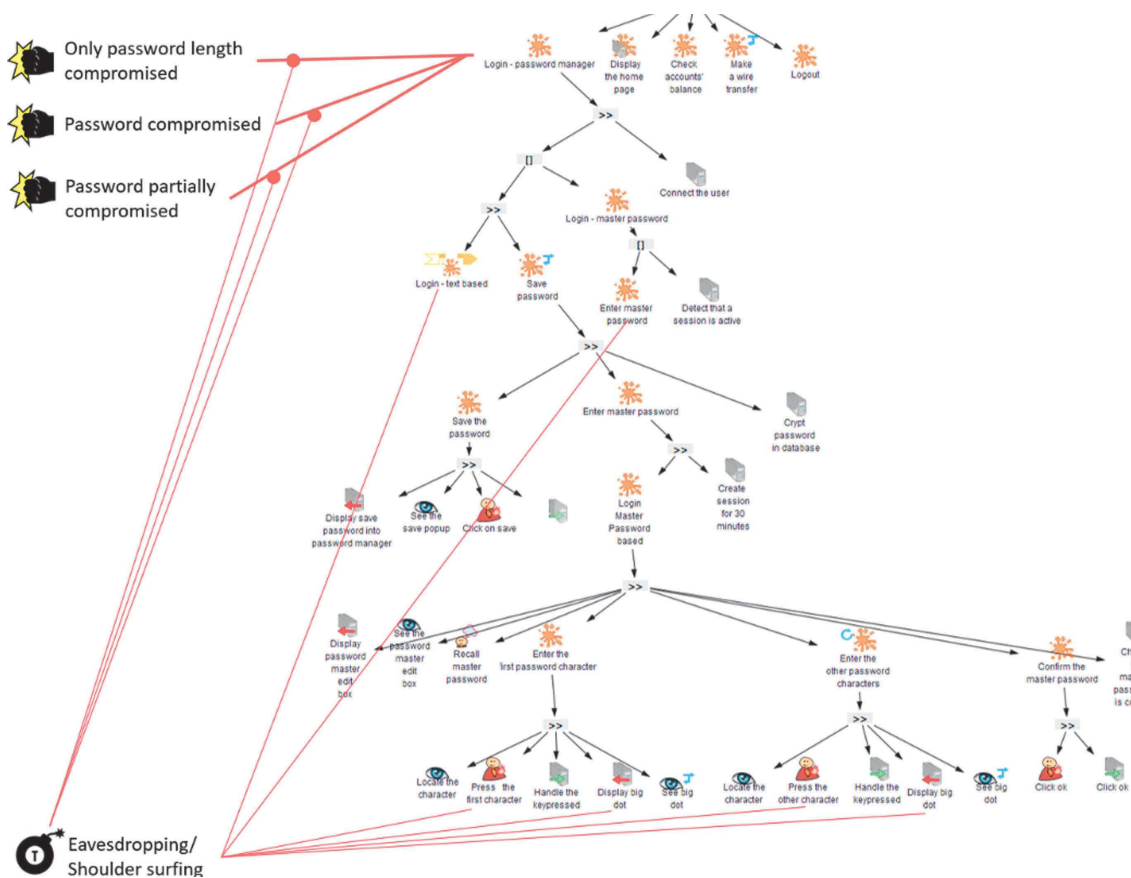
Fig. 11. Attack Tree of the Firefox Password Manager Mechanism

### Integrated Model of Tasks, Threats and Effects for the Password Manager

In addition to the description of user tasks, the task models in Fig. 12 and Fig. 13 present the possible threats and their associated effects. In Fig. 12, “Login - text based”, “Save password” and “Enter master password” are abstract tasks concerned by the same threats (keylogging, shoulder surfing, and video recording attacks). The effects of these threats can be that only length password compromised, or that password partially/entirely compromised. Figure 13 concerns the shoulder surfing threat.

In Fig. 13 the representation of objects, knowledge and information has been filtered out so that the reader can focus on threats and effects. The shoulder surfing threat may





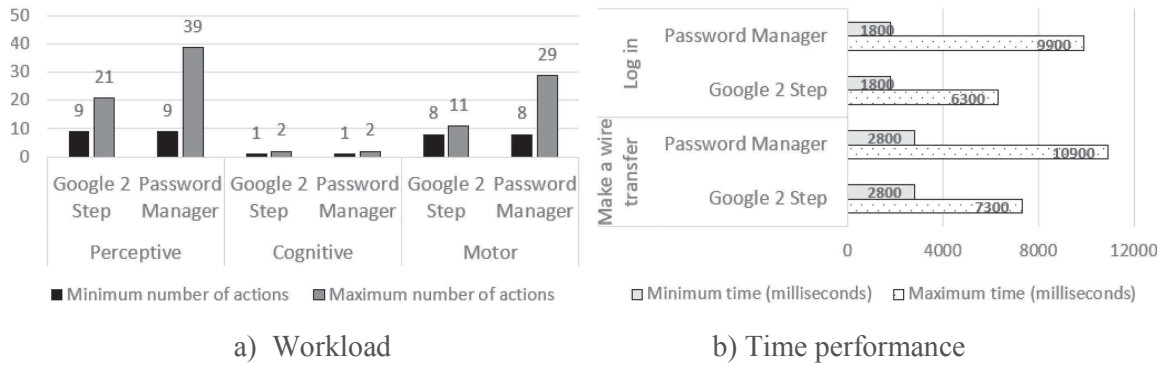
**Fig. 13.** Login with password manager task model (refinement of “Save password” task)

4. Calculation (from the attack tree) of the minimum and maximum number of attacks required to compromise user password, as well as interdependencies between these attacks (represented by AND and SAND operators).

These steps must be performed for each authentication mechanisms that have to be analyzed. The output of these steps is the predicted cost of the authentication mechanism in terms of effectiveness and efficiency, as well as the expected benefits of the authentication mechanisms in terms of security. The last step of the analysis is to compare the costs and benefits of the two authentication mechanisms.

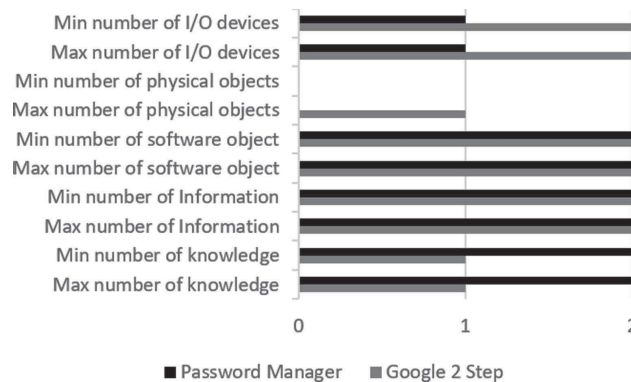
From the task models of the usage of the two authentication mechanisms, we see that the planned user tasks with both mechanisms lead to reach the login sub-goal. Both authentication mechanisms are thus effective. The task models enable to identify systematically all the possible sequences of actions. From these possible sequences of actions, we calculate the minimum and maximum amount of perceptive, cognitive and motor tasks. The results are presented in Fig. 14 a. For the calculation of the minimum amount of tasks, we take as a basis the fact that the minimum password length required is 8 characters. And for the maximum amount of tasks, we chose to take a password length of 9 for the calculation. In Fig. 14 a), we can see that the minimum amount of actions is similar between the two authentication mechanisms. However, we can also see that the situation is completely different when the whole set of actions has to be performed for the Firefox Password Manager authentication mechanism, in case where the password

has to be saved using the master password. In this case the number of perceptive and motor actions is more than 3 times higher. This conclusion also arises when calculating the minimum and maximum amount of time taken to login with the two authentication mechanisms, as presented in Fig. 14 b. The calculation of the time costs has been done using the Keystroke Level Model [13].



**Fig. 14.** Results of the predictive assessments of user workload and user time performance with Google 2 Step and Password Manager Mechanisms for reaching the sub-goal of “Login”

Figure 15 presents the minimum and maximum number of data, objects and devices required to use the authentication mechanisms. For the Google 2 Step mechanism, two I/O devices are at least always required, whereas only one device is always required to use the Firefox Password Manager. It is also highlighted that more knowledge is required to use the Firefox Password Manager. The amount of software object and information required is similar for both authentication mechanisms. The two authentication mechanisms cover the same types of attacks: Eavesdropping (key logging, video recording, shoulder surfing), brute force, phishing. There are thus no differences between the two authentication mechanisms in terms of types of attacks covered. However, there are differences in terms of complexity to achieve an attack.

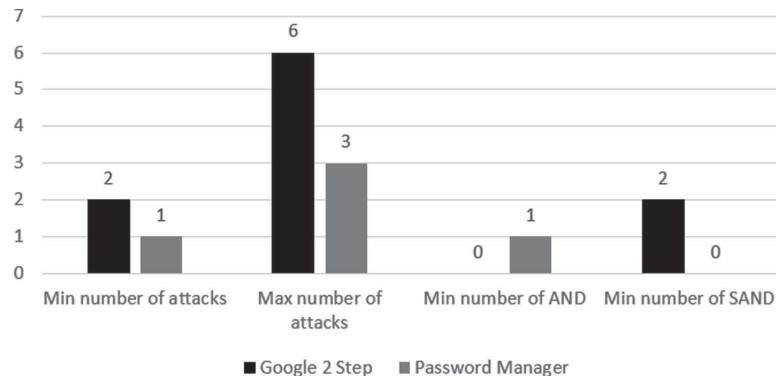


**Fig. 15.** Comparison of the required data, objects and devices

Figure 16 presents the minimum and maximum number of attacks that have to be performed to compromise the user password, as well as the minimum number of operators AND and SAND in the attack tree. These values provide support to analyze



the complexity of the authentication mechanisms. For both authentication mechanisms, several attacks are required to compromise user password (at least one AND or SAND element in the attack trees), which makes the hacking process not easy. However, the Google 2 Step is more complicated as at a specific temporal ordering is required for performing the attacks (at least two SAND elements in the attack tree).



**Fig. 16.** Comparison of the attack trees

To conclude this analysis, the Google 2 Step seems to be more usable for entering one password than the Firefox Password Manager. It also appears to be and more secure as to compromise the password, the attack has to be more complex than for the Firefox Password Manager.

We have shown the comparison of the two authentication mechanisms for a user that logs in one time. But, users are everyday login several times with different passwords they have to remember and recall. Users have an average of 25 accounts requiring identification [21], and as capability of users to remember and recall multiple passwords is questioned [7], the usage of password manager brings a usability benefit as the user needs to remember and recall a single password once the password manager has been configured. From a security point of view, the impact of an attack on the Firefox Password Manager is more important because the attacker may compromise all the user passwords.

## 5 Conclusion

The goal of this paper was to present two complementary modelling techniques used to analyze jointly usability and security. We presented how the modelling techniques were used to compare usability and security of two authentication mechanism: Google 2 Step and Firefox Password Manager. From a security perspective, the approach supports the explicit and systematic analysis of threat coverage and of authentication mechanism complexity. From a usability perspective, the presented modelling approach supports the explicit and systematic analysis of the effectiveness and efficiency contributing factors. Compared to user testing techniques, a task modelling based approach enables to analyze the possible tasks for several activities to be performed on authentication mechanisms (e.g. to learn to use, to configure, to reset the password...) and to compare authentication mechanisms without performing empirical evaluations and involving users. However, the



proposed approach is compatible with empirical user testing and the measures of users' satisfaction are also relevant for the analysis [50]. In the area of security measurement of user's subjective perception is critical as it is the main mean to assess trust in the security mechanisms as demonstrated in [38].

The presented work is currently extended to assess more sophisticated authentication mechanisms such as EEVEHAC that ensure secure authentication on potentially compromised devices. Beyond, as it was performed in [35] with "standard" task models, we are working on the connection of the integrated task models to the actual authentication mechanism to do predictive performance evaluation [43] and to support automatic assessment of effectiveness. Beyond, such coupling (as it encompasses visual and behavioral aspects [5]) would allow precise description of interaction techniques (such as multi-touch ones [27]) and their impact on both usability and security, beyond the authentication mechanism itself.

Lastly, due to space constraints, the paper has only presented the verification of identity sub task of authentication. However, other sub tasks such as enrolment (creation of account), modification of credential and revocation can be captured in a similar way (including both tasks and threats) using the notation presented in the paper.

## References

1. Adams, A., Sasse, M.A.: Users are not the enemy. *Commun. ACM* **42**(12), 40–46 (1999)
2. Alarifi, A., Alsaleh, M., Alomar, N.: A model for evaluating the security and usability of e-banking platforms. *Computing* **99**(5), 519–535 (2017). <https://doi.org/10.1007/s00607-017-0546-9>
3. Alshamari, M.: A review of gaps between usability and security/privacy. *Int. J. Commun. Network Syst. Sci.* **9**, 413–429 (2016)
4. Balfanz, D., Durfee, G., Smetters, D.K., Grinter, R.E.: In search of usable security: five lessons from the field. *IEEE Secur. Priv.* **2**(5), 19–24 (2004)
5. Bastide, R., Palanque, P.: A visual and formal glue between application and interaction. *J. Vis. Lang. Comput.* **10**(5), 481–507 (1999). ISSN 1045-926X
6. Ben-Asher, N., Meyer, J., Möller, S., Englert, R.: An experimental system for studying the tradeoff between usability and security. In: *International Conference on Availability, Reliability and Security, Fukuoka, 2009*, pp. 882–887 (2009)
7. Bonneau, J., Schechter, S.: Towards reliable storage of 56-bit secrets in human memory. In: *USENIX Security Symposium* (2014)
8. Bonneau, J., Herley, C., van Oorschot, P.C., Stajano, F.: The quest to replace passwords: a framework for comparative evaluation of web authentication schemes. In: *2012 IEEE Symposium on Security and Privacy*, pp. 553–567 (2012)
9. Brainard, J., Juels, A., Rivest, R.L., Szydlo, M., Yung, M.: Fourth-factor authentication: somebody you know. In: *Proceedings of the 13th ACM CCS 2006*, pp. 168–178. ACM (2006)
10. Braz, C., Seffah, A., M'Raihi, D.: Designing a trade-off between usability and security: a metrics based-model. In: Baranauskas, C., Palanque, P., Abascal, J., Barbosa, S.D.J. (eds.) *INTERACT 2007*. LNCS, vol. 4663, pp. 114–126. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-74800-7\\_9](https://doi.org/10.1007/978-3-540-74800-7_9)
11. Brooke, J.: SUS - A quick and dirty usability scale (2006)
12. Brostoff, S., Sasse, M.A.: Are passfaces more usable than passwords? In: *A Field Trial Investigation BCS HCI Conference, People and Computers XIV—Usability or Else!* (2000)

13. Card, S.K., Moran, T.P., Newell, A.: The model human processor: an engineering model of human performance. In: *Handbook of Perception and Human Perf.*, pp. 1-35 (1986)
14. Chiasson, S., Biddle, R.: Issues in user authentication. In *CHI Workshop Security User Studies Methodologies and Best Practices*, April 2007
15. Clark, R.M., Hakim, S. (eds.): *Cyber-Physical Security: Protecting Critical Infrastructure at the State and Local Level*, vol. 3. Springer, Cham (2016). <https://doi.org/10.1007/978-3-319-32824-9>
16. De Cristofaro, E., Du, H., Freudiger, J., Norcie, G.: A comparative usability study of two-factor authentication. In: *Proceedings of the Workshop on Usable Security (USEC)* (2014)
17. Fahssi, R., Martinie, C., Palanque, P.: Enhanced task modelling for systematic identification and explicit representation of human errors. In: Abascal, J., Barbosa, S., Fetter, M., Gross, T., Palanque, P., Winckler, M. (eds.) *INTERACT 2015. LNCS*, vol. 9299, pp. 192–212. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-22723-8\\_16](https://doi.org/10.1007/978-3-319-22723-8_16)
18. Faily, S., Fléchain, I.: Finding and resolving security misusability with misusability cases. *Requirements Eng.* **21**(2), 209–223 (2016)
19. Fayollas, C., Martinie, C., Navarre, D., Palanque, P.: A generic approach for assessing compatibility between task descriptions and interactive systems: application to the effectiveness of a flight control unit. *i-com*, 14(3), 170–191 (2015)
20. Firefox password manager. <https://support.mozilla.org/en-US/kb/password-manager-remember-delete-edit-logins>
21. Florencio, D., Herley, C.: A large-scale study of web password habits. In: *Proceedings of the WWW Conference 2007*, pp. 657–666. ACM Press (2007)
22. Fraile, M., Ford, M., Gadyatskaya, O., Kumar, R., Stoelinga, M., Trujillo-Rasua, R.: Using attack-defense trees to analyze threats and countermeasures in an ATM: a case study. In: Horkoff, J., Jeusfeld, Manfred A., Persson, A. (eds.) *PoEM 2016. LNBIP*, vol. 267, pp. 326–334. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-48393-1\\_24](https://doi.org/10.1007/978-3-319-48393-1_24)
23. Golla, M., Bailey, D.V., Dürmuth, M.: “I want my money back!” Limiting Online Password-Guessing Financially. In: *SOUPS 2017* (2017)
24. Google 2-step Verification. <https://www.google.com/landing/2step/>. Accessed May 2020
25. Habib, H., et al.: User behaviors and attitudes under password expiration policies. In: *USENIX Security Symposium 2018*, pp. 13–30 (2018)
26. Halunen, K., Häikiö, J., Vallivaara, V.A.: Evaluation of user authentication methods in the gadget-free world. *Pervasive Mob. Comput.* **40**, 220–241 (2017)
27. Hamon, A., Palanque, P., Silva, J.-L., Deleris, Y., Barboni, E.: Formal description of multi-touch interactions. In: *5th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2013)*, pp. 207–216 (2013)
28. He, W., et al.: Rethinking access control and authentication for the home Internet of Things (IoT). In: *USENIX Security Symposium*, pp. 255–272 (2018)
29. ISO. “ISO 9241-11 :2018”. ISO. International Organization for Standardization. <https://www.iso.org/standard/63500.html>
30. ISO. “ISO 9241-210:2019”. ISO. International Organization for Standardization. Accessed 17 Feb 2020. <https://www.iso.org/standard/77520.html>
31. ISO/IEC 27000:2018 Information technology—Security techniques—Information security management systems (2018)
32. Kainda, R., Fléchain, I., Roscoe, A.W.: Security and usability: analysis and evaluation. In: *2010 International Conference on Availability, Reliability and Security*, pp. 275–282 (2010)
33. Launius, S.M.: Evaluation of Comprehensive Taxonomies for Information Technology Threats. *SysAdmin, Audit, Network and Security (SANS)* (2018)

34. Martinie, C., Navarre, D., Palanque, P., Fayollas, C.: A generic tool-supported framework for coupling task models and interactive applications. In: Proceedings of the 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2015). ACM DL, pp. 244–253 (2015)
35. Martinie C., Navarre D., Palanque P., Fayollas, C.: A generic tool-supported framework for coupling task models and interactive applications. In: 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2015). ACM DL, pp. 244–253 (2015)
36. Martinie, C., Palanque, P., Winckler, M.: Structuring and Composition Mechanisms to Address Scalability Issues in Task Models. In: Campos, P., Graham, N., Jorge, J., Nunes, N., Palanque, P., Winckler, M. (eds.) INTERACT 2011. LNCS, vol. 6948, pp. 589–609. Springer, Heidelberg (2011). [https://doi.org/10.1007/978-3-642-23765-2\\_40](https://doi.org/10.1007/978-3-642-23765-2_40)
37. Martinie, C., Palanque, P., Bouzekri, E., Cockburn, A., Canny, A., Barboni, E.: Analysing and demonstrating tool-supported customizable task notations. PACM Hum.-Comput. Interact. **3** (2019). EICS, Article 12, 26 pages
38. Merdenyan, B., Petrie, H.: Perceptions of risk, benefits and likelihood of undertaking password management behaviours: four components. In: Lamas, D., Loizides, F., Nacke, L., Petrie, H., Winckler, M., Zaphiris, P. (eds.) INTERACT 2019. LNCS, vol. 11746, pp. 549–563. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-29381-9\\_34](https://doi.org/10.1007/978-3-030-29381-9_34)
39. Micallef, N., Gamagedara Arachchilage, N.A.: A Gamified Approach to Improve Users' Memorability of Fall-back Authentication. SOUPS 2017 (2017)
40. Mihajlov, M. Jerman-Blazič, B., Josimovski, S.: A conceptual framework for evaluating usable security in authentication mechanisms - usability perspectives. In: 2011 5th International Conference on Network and System Security, Milan, 2011, pp. 332–336 (2011)
41. Nishihara, H., Kawanishi, Y., Souma, D., Yoshida, H.: On validating attack trees with attack effects. In: Casimiro, A., Ortmeier, F., Bitsch, F., Ferreira, P. (eds.) SAFECOMP 2020. LNCS, vol. 12234, pp. 309–324. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-54549-9\\_21](https://doi.org/10.1007/978-3-030-54549-9_21)
42. Ortega-Garcia, J., Bigun, J., Reynolds, D., Gonzalez-Rodriguez, J.: Authentication gets personal with biometrics. IEEE Signal Process. Mag. **21**(2), 50–62 (2004)
43. Palanque, P., Barboni, E., Martinie, C., Navarre, D., Winckler, M.: A model-based approach for supporting engineering usability evaluation of interaction techniques. In: 3rd ACM SIGCHI Symposium on Engineering interactive Computing Systems (EICS 2011), pp. 21–30 (2011)
44. Palanque, P., Basnyat, S.: Task patterns for taking into account in an efficient and systematic way both standard and erroneous user behaviors. In: IFIP 13.5 Working Conference on Human Error, Safety and Systems Development (HESSD), pp. 109–130. Kluwer Academic Publishers (2004)
45. Petsas, T., Tsirantonakis, G., Athanasopoulos, E., Ioannidis, S.: Two-factor authentication: is the world ready? quantifying 2FA adoption. In: Proceedings of the Eighth European Workshop on System Security (EuroSec 2015). ACM, Article 4, 1–7 (2015)
46. Raza, M., Iqbal, M., Sharif, M., Haider, W.: A survey of password attacks and comparative analysis on methods for secure authentication. World Appl. Sci. J. **19**(4), 439–444 (2012)
47. Rosson, M.B., Carroll, J.M.: Usability Engineering: Scenario-Based Development of Human-Computer Interaction. Elsevier (2001)
48. Sasse, A.: Computer security: anatomy of a usability disaster, and a plan for recovery. In: Proceedings of CHI 2003 Workshop on HCI and Security Systems. Fort Lauderdale, Florida (2003)
49. Schneier, B.: Attack Trees. Dr. Dobb's J., December 1999

50. Seiler-Hwang, S., Arias-Cabarcos, P., Marín, A., Almenares, F., Díaz-Sánchez, D., Becker, C.: “I don’t see why I would ever want to use it”: analyzing the usability of popular smartphone password managers. In: Proceedings of the ACM SIGSAC CCS 2019, pp. 1937–1953. ACM (2019)
51. Weaver, A.C.: Biometric authentication. *Computer* **39**(2), 96–97 (2006)