Enabling Automatic Password Change in Password Managers Through Crowdsourcing
Short Paper

Peter Mayer[1], Hermann Berket, and Melanie Volkamer[1][2]

[1] Technische Universität Darmstadt, Department of Computer Science, SECUSO - Security, Usability, Society
`firstname.lastname@secuso.org`
[2] Karlstad University, Privacy and Security Research Group

Abstract. Password managers can immensely increase password security, for laymen and experts alike. One reason for the low adoption is a lack of automatic password change. In this paper, we describe our work on extending password managers towards automatic password change on the web. The main idea is that users can teach the password managers how to change their password on a specific website using a recorder module and share this information (blueprints) with other users, effectively crowdsourcing the creation of these blueprints. Parts of the proposal were implemented as an add-on for the Firefox browser and evaluated in a user study. Our results show that our proposal is viable and usable.

Keywords: automatic password change, password managers, crowdsourcing

1 Introduction

The usage of password managers can be highly beneficial to the security of passwords, for experts and laymen alike [9]. They enable their users to choose a different secure, random password for each account. One reason for the low adoption is that password managers nowadays do not really offer the functionality to change passwords automatically [9]. However, this functionality is already needed, when users first start using a password manager. In such cases, he or she might have a potentially large amount of legacy, self-created passwords [9]. Furthermore, users may also want to change their passwords later on, including when a verifier (e.g. webservice) is compromised, because they shared the respective password with someone or they just feel it is not secure enough anymore. The wide variation of password change processes across the web renders automated change difficult to implement for developers of password managers [7].

In this paper, we present a proposal for an extension to existing password managers offering automatic password change on arbitrary websites. The main idea is that users can record the procedure of how to change a password on a specific website as a blueprint and share this information with other users.

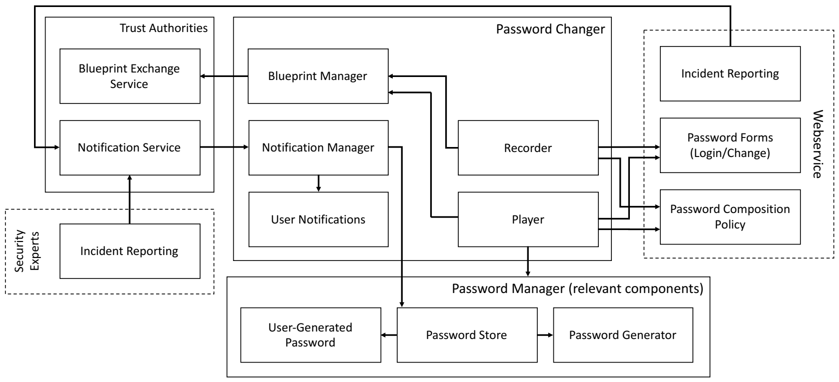


Fig. 1: Our proposed concept for automatic password change on the web. All arrows suggest a *uses* relation, e.g. the player uses the blueprints from the blueprint manager, the password manager (to get the current password and request a newly generated one) and the password form to change the password.

Thereby, it is possible to effectively crowdsource the effort needed to support a wide range of websites. In the same way, our proposal includes crowdsourced sharing of information regarding security breaches, so users can react in a timely fashion.

The main functionality of our proposal was implemented as a Firefox add-on. To evaluate the practicality of our add-on, we conducted a user study. Participants used both, the recording as well as the actual password changing features. While getting some feedback for improvements, the results indicate that crowdsourced automatic password change is both, viable and usable.

2 Description of our Proposed Concept

Some password managers already support automatic password change (e.g. LastPass [3], Dashlane [2]). However, the selection of supported websites is limited and caters mostly to the U.S. market. Users have little to no influence on what websites are supported, even if they visit them frequently.

The primary goal in designing our proposal of automatic password change is enhancing the current approaches. We strive for a solution which allows supporting the websites users actually have accounts at, no matter how niche they might be. Our proposed concept is based on existing password managers and introduces two new building blocks which are described in the following subsections (see Figure 1 for an overview).

2.1 Password Changer

The password changer is the core part of our proposal. It comprises five components. The first component is the *blueprint manager*. It represents the repository for the blueprints needed by the user and manages access to it. Furthermore, it allows sharing blueprints among users of our proposed password manager extension through the blueprint exchange service (see section 2.2). The second component is the *recorder*. This component allows to record the procedure of how to change the password on a specific website while users perform the change once manually. The recorder then stores a transcript of the necessary actions (e.g. websites to visit, forms to fill, etc.) as a blueprint. Ideally, the website provides the enforced password composition policy in a standardised way as proposed by [7, 8]. If that is not the case, the recorder asks the recording user to provide the policy for inclusion in the blueprint. The third component is the *player*. It represents the core functionality from the user's point of view: the automatic change of passwords using the blueprints. The user only needs to decide whether to use a self-generated or randomly generated password. The automatic change can then either take place in the background (without disturbing the user once started) or be displayed to the user, depending on the user's preference regarding transparency. If the blueprint does not contain the password policy, the player requests it from the corresponding website according to [7, 8]. In case of a user-generated password, a dialogue asking for the preferred password is displayed. This dialogue also includes the password policy.

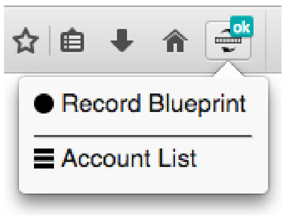
The last two components are the *notification manager* and the *user notifications*. One of the primary benefits of automatic password change is that users can respond to breaches on the verifier side with very low effort. We propose the usage of a notification service to inform users of noteworthy events such as breaches at webservices in order to minimise the time frame an attacker has to exploit such an incident. Security experts as well as webservice providers suffering from a security incident are supposed to use this channel. Some password managers already include similar notification services (e.g. 1password [1]).

2.2 Trust Authorities

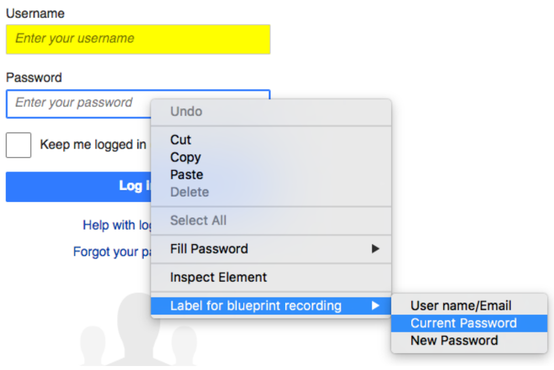
Due to the sensitive nature of the distributed data and potential security risks associated with it, we believe a manual screening process of the crowdsourced information is essential. The natural entity assuming this role would be the developer of the respective password manager. However, our concept explicitly allows multiple *trust authorities*. Users can then selectively subscribe to those authorities they prefer (analogous to e.g. filter lists used in ad blockers). The trust authorities represent the regulation system for both blueprints and notifications. Correspondingly, the trust authorities can offer two services: a *blueprint exchange service* and a *notification service*. In particular, we propose that the trust authorities should digitally sign all crowdsourced information to ensure its authenticity (analogous to e.g. app stores).

3 Prototype Implementation

To evaluate the practicability of our proposal, we implemented the core components in a prototype Firefox add-on extending the built-in password manager's



(a) Toolbar-button and the menu



(b) Context menu for labelling forms during the recording

Fig. 2: The visual design of our prototype add-on.

functionality with the following components: the *recorder*, *player*, and *blueprint manager*. Note, we additionally implemented a password generator because the built-in password manager does not include one.

3.1 Description and Encountered Challenges

All functionality is accessible via a button in the toolbar, which also indicates notifications from the notification service (see Figure 2a). The user initiates the recording in the menu accessible through this button. While implementing the corresponding functionality, we encountered the *first challenge*: Automatic detection of the different form fields (i.e. user name, current password, new password) across different websites can be unreliable due to the variety of technologies and designs in use (an issue already described in [7]). To still record reliably, the add-on requires the user to label the form elements using the context menu as shown in Figure 2b, to indicate which information needs to be inserted into which form field. Once labelled, the corresponding form fields are highlighted in yellow to give immediate feedback to the user.

All recordings should include the logout procedure, so that users changing their password based on the blueprint do not remain logged in after the password change. Here, we encountered the *second challenge*: The logout procedure differs from website to website and its completion can therefore not be automatically detected by the add-on. In our prototype add-on, we addressed this challenge by requiring the user to manually stop the recording, once all actions involved in the password change are completed.

The actual password change is initiated through the account list option in the toolbar-button menu. It directs the user to a list of all managed accounts, where users can initiate a change for any account in the list using either a password generated by the password manager or themselves. If for an account no blueprint is available, users are asked whether they want to record a blueprint for the respective website. Upon completion of the automatic password change using a generated password, the user is informed using an alert pop-up. When a user chooses to change the password manually instead, only the navigation actions included in the blueprint are executed. The forms (i.e. login and actual password change) need to be filled by the user.

While implementing the player, we faced *two more challenges*: Firstly, technical limitations in the Firefox add-on API rendered it impossible to execute a fully automatic password change entirely in the background. Thus, the add-on executes the blueprint in a normal browser tab, visible to the user. Secondly, various websites (e.g. GitHub) implement abuse detection mechanisms. Such mechanisms can be triggered, when e.g. form fields are filled and submitted to fast. Currently, the add-on simply introduces a delay of 2 seconds after each step in the blueprint. An alternative would be to also record timings and include them in the blueprints.

3.2 User study

To investigate the usability and practicality of our prototype implementation, we conducted a small user study employing a think aloud methodology. The participants were told that the purpose is to evaluate an add-on developed at the computer science department. After having signed a consent form, participants received a short tutorial. Then, they were asked to record a blueprint and initiate an automatic password change using a password generated by the add-on. All tasks were performed on a laptop provided by us, with Firefox and the add-on pre-installed. After all tasks were completed, participants were asked to fill out a System Usability Scale (SUS) questionnaire [6].

Eight participants from both, IT and non-IT backgrounds, and of different ages took part in our study. With a mean of 82.5, the average SUS score indicates that the overall experience is a positive one (a mean above 68 is considered "as above average usability", according to [5] a good to excellent usability score). However, due to the low number of participants this value can only serve as a rough indication. We also analysed the comments made while using the add-on: The most prevalent issue voiced by some participants was that recording of the blueprints required labelling the password forms. Participants stated that this was unintuitive and perceived it as unnecessary workload. Furthermore, aspects regarding the trust in such an extension were voiced by security expert participants. The arising question was whether the blueprints might be manipulated while being stored on the hard disk. The raised concerns are rather related to the information participants had about the add-on than about the actual usability. However, the concerns show how important a proper presentation of the add-on's features and capabilities is.

4 Conclusion

In this paper, we presented our concept enabling automatic password change on the web using crowdsoursing to support a wide range of websites. We briefly introduced the prototype add-on we implemented and the challenges we faced during its implementation. From a small user study we learnt how important a proper presentation of such an add-on would be. In particular, our next step will be implementing the remaining components and developing an appropriate presentation. We plan to evaluate the entire concept afterwards. To motivate participation in the crowdsourcing activities (i.e. creation of new blueprints, improvements to existing blueprints, or rightful reporting of security incidents), we will discuss how to use incentive mechanisms such as studied in [4]. Overall, we believe that our proposal can increase adoption of password managers. In particular, our proposal allows users to change existing passwords to secure, random ones in an easy and usable fashion.

5 Acknowledgements

This work has been developed within the project 'KMU AWARE' which is funded by the German Federal Ministry for Economic Affairs and Energy under grant no. BMWi-VIA5-090168623-01-1/2015. The authors assume responsibility for the content.

References

1. 1password watchtower. https://watchtower.agilebits.com/. Accessed: 2016-03-15.
2. Dashlane launches password changer. https://blog.dashlane.com/dashlane-launches-password-changer-automatically-changes-passwords-seconds-one-click-solution-constant-security-breaches-press-release/. Accessed: 2016-03-16.
3. Introducing auto-password changing with lastpass. https://blog.lastpass.com/2014/12/introducing-auto-password-changing-with.html/. Accessed: 2016-03-16.
4. A. Anderson, D. Huttenlocher, J. Kleinberg, and J. Leskovec. Steering user behavior with badges. In *WWW*, pages 95–106. ACM, 2013.
5. A. Bangor, P. Kortum, and J. Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *Journal of usability studies*, 4(3):114–123, 2009.
6. J. Brooke et al. SUS-A quick and dirty usability scale. In *Usability evaluation in industry*. Taylor & Francis, 1996.
7. G. J. Frank Stajano, Max Spencer and Q. Stafford-Fraser. Password-manager friendly (pmf): Semantic annotations to improve the effectiveness of password managers. In *International Conference on Passwords*. Springer LNCS, 2014.
8. M. Horsch, M. Schlipf, J. Braun, and J. Buchmann. Password requirements markup language. In *Australasian Conference on Information Security and Privacy*, pages 426–439. Springer LNCS, 2016.
9. E. Stobert and R. Biddle. Expert Password Management. In *International Conference on Passwords*, pages 3–20. Springer LNCS, 2015.