Passlab: A Password Security Tool for the Blue Team

Saul Johnson[0000−0001−9876−3775]

Teesside University, Middlesbrough, UK
saul.johnson@tees.ac.uk

Abstract. If we wish to compromise some password-protected system as an attacker (i.e. a member of the *red team*), we have a large number of popular and actively-maintained tools to choose from in helping us to realise our goal. Password hash cracking hardware and software, online guessing tools, exploit frameworks, and a wealth of tools for helping us to perform reconnaissance on the target system are widely available. By comparison, if we wish to defend a password-protected system against such an attack (i.e. as a member of the *blue team*), we have comparatively few tools to choose from. In this research abstract, we present our work to date on Passlab, a password security tool designed to help system administrators take advantage of formal methods in order to make sensible and evidence-based security decisions using a clean and intuitive user interface.

Keywords: software tool · threat analysis · policy · guessing

1 Introduction

When it comes to making decisions about the most appropriate password security policy for a system, the tendency has traditionally been to take a "common sense" approach. For example, it stands to reason intuitively that forcing a user to include at least one number in their password will make that password harder to guess and therefore more secure. Applying formal methods in order to quantify this increase in security, however, is seldom part of the decision-making process, leading to widely varying password security policies born from equally variable intuitions about which factors contribute to their security. Previous work finds that the password composition policy in place on a system has little to no correlation with the value of the assets it protects [7]. We expect that tightening legislation around data protection (in Europe in particular [4]) will encourage industry to invest in tools that offer the ability to make data-driven password security policy decisions.

The expected contribution is Passlab, an integrated environment that will allow system administrators without a background in formal methods to make informed password security decisions, formally reason about password composition policies and extract correct-by-construction software for enforcing them.

2 Literature Review

We draw on work exploring the distribution of passwords by Malone and Maher [11] and Wang et al. [12] in order to derive equations such as that in Figure 1 (see Section 3.1). The model of password composition policies presented by Blocki et al. in [2] has shown promise thus far as a general low-level representation that allows us to encode password composition policies for a wide variety of password quality checking software. Work on attack-defence trees by Kordy et al. [9] offers us a way to allow system administrators without a formal methods background to model password guessing attacks and password composition policies to mitigate them in an intuitive and visual way. The Coq interactive theorem proving software [1] and its code extraction capabilities [10] offer us a means to create usable, formally-verified password composition policy enforcement software.

3 Progress to Date

In this section, we present progress to date on Passlab itself, and on work that forms the foundations of the formal methods it employs.

3.1 Data-Informed Lockout Policies

A *lockout policy* is a restriction on the number of times a user can incorrectly enter their password before their account is locked down, requiring additional authentication via some other mechanism in order to reinstate their ability to log in. This offers strong protection against so-called *online* password guessing attacks (i.e. attacks against the live service) but in turn introduces a denial-of-service vulnerability. If an attacker wants to prevent a user from accessing their account, they need only attempt to access it with the wrong password enough times that the lockout policy is triggered. This motivates us to search for formal methods to derive the maximum number of incorrect login attempts we can grant a user while guaranteeing that the probability of guessing attack success is kept below a specified threshold in the worst case.

Figure 1 shows a render of the Passlab user interface as it fits a power-law equation that maps the probability of a password being a correct guess (x) to its rank (y) in a large password data dump (in this case, the RockYou data set [3]). The software allows users to visually compose data analysis tasks such as that illustrated in Figure 1. This draws on previous research, which finds that user-chosen passwords tend to follow Zipf's law in the general case [11,12]. That is, the frequency (and therefore probability) of a password is inversely proportional to its position in the data set, when ranked by frequency. In this case, it is possible to use this equation to calculate that, even if an attacker knew the most common 8 passwords on our system, if they selected an account at random and tried these 8 passwords they would have a probability of successfully gaining entry to that account of less than 0.02.

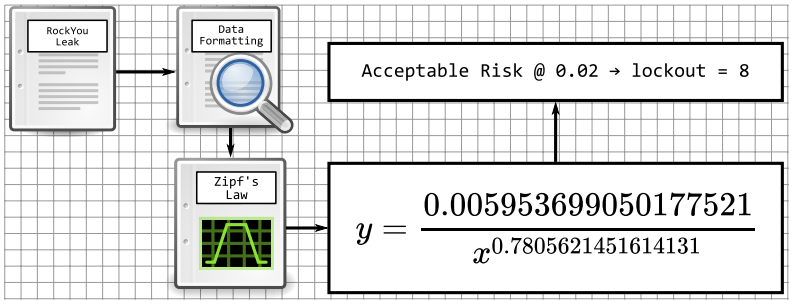

Fig. 1. A mock-up of the Passlab user interface. A raw data source node loads a raw password data dump (top left) which is then formatted (top centre) to convert it to a CSV file. After formatting, the data enters a Zipf model node (bottom centre) which computes a power-law equation to approximate guess success probability from password guessing order (bottom right).

3.2 Interactive Security Policy Building

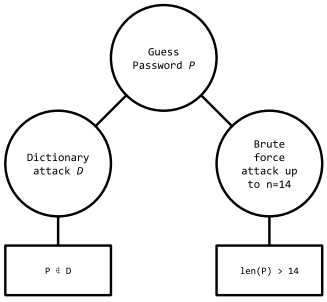

Fig. 2. An abstract example of an attack-defence tree (ADTree) [9] from which a password composition policy might be synthesised.

Passlab will include an "interactive security policy builder" from within which a system administrator can model a password guessing attack and its mitigation measures as an attack-defence tree (ADTree) [9] and synthesise password composition policy enforcement software (see Section 3.3). Figure 2 shows how such an ADTree might look, with policy-level mitigations for each mode employed by the password guessing attack in place. The attacker has a goal of guessing a password and, to try and achieve this, employs a bimodal guessing attack—a dictionary attack using dictionary D and a brute-force attack of passwords of length up to 14. To mitigate this, we add defence nodes to ensure that the password is not contained in D and that its length is greater than 14.

After the system administrator builds a security policy in this way, we would like them to be able to perform further reasoning from outside that environment. Work on this is ongoing in the form of *Skeptic*, our Coq framework for reasoning about password composition policies [8], which has already shown promise in formally verifying that certain password composition policies confer immunity to certain attacks, for example. Creating software that combines analysis of large data sets with formal verification for usable security continues to be one of the main challenges in our research.

3.3 Correct-by-Construction Enforcement Software

We have previous work [6] on creating certified password composition policy enforcement software, implemented from within the Coq proof assistant [1] and

extracted to Haskell [10]. The extracted Haskell is then compiled into a pluggable authentication module readily usable from a real Linux system. Building on this work, we created *Serenity* [5], a DSL for building correct-by-construction password quality checking software that employs formal verification to ensure that password composition policies built using it are correct. A small user study has provided encouraging preliminary results indicating that users find it easier to express their desired password composition policy using the Serenity DSL than they do using more traditional password composition policy enforcement software.

References

1. Bertot, Y., Castéran, P.: Interactive theorem proving and program development – Coq'Art: the calculus of inductive constructions. Springer Science & Business Media (2013)
2. Blocki, J., Komanduri, S., Procaccia, A., Sheffet, O.: Optimizing password composition policies. In: Proceedings of the Fourteenth ACM Conference on Electronic Commerce. pp. 105–122. EC '13, ACM, New York, NY, USA (2013). https://doi.org/10.1145/2482540.2482552
3. Cubrilovich, N.: Rockyou hack: From bad to worse — techcrunch. `https://tcrn.ch/2IVkCpP` (Dec 2009), (Accessed on 04/10/2019)
4. European Parliament: Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). Official Journal of the European Union **59**, 1–88 (2016)
5. Ferreira, J.F., Johnson, S.A., Mendes, A., Cordry, J.: Serenity: A domain-specific language for certified password quality (2019), to be submitted.
6. Ferreira, J.F., Johnson, S.A., Mendes, A., Brooke, P.J.: Certified password quality. In: Polikarpova, N., Schneider, S. (eds.) Integrated Formal Methods. pp. 407–421. Springer International Publishing, Cham (2017)
7. Florêncio, D., Herley, C.: Where do security policies come from? In: Proceedings of the Sixth Symposium on Usable Privacy and Security. pp. 10:1–10:14. SOUPS '10, ACM, New York, NY, USA (2010). https://doi.org/10.1145/1837110.1837124
8. Johnson, S.A., Ferreira, J.F., Mendes, A., Cordry, J.: Skeptic: A practical framework for formal reasoning about password composition policies (2019), to be submitted.
9. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Foundations of attack–defense trees. In: Degano, P., Etalle, S., Guttman, J. (eds.) Formal Aspects of Security and Trust. pp. 80–95. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
10. Letouzey, P.: Extraction in Coq: An overview. In: Conference on Computability in Europe. pp. 359–369. Springer (2008)
11. Malone, D., Maher, K.: Investigating the distribution of password choices. In: Proceedings of the 21st International Conference on World Wide Web. pp. 301–310. WWW '12, ACM, New York, NY, USA (2012). https://doi.org/10.1145/2187836.2187878
12. Wang, D., Cheng, H., Wang, P., Huang, X., Jian, G.: Zipfs law in passwords. IEEE Transactions on Information Forensics and Security **12**(11), 2776–2791 (Nov 2017). https://doi.org/10.1109/TIFS.2017.2721359