Security Key Authentication for Web Applications

CS588 - Networked Distributed System Security | Professor J. Solworth

Davide Giacomini
dept. of Engineering
University of Illinois at Chicago (UIC)
Chicago (IL), USA
giacomini.davide@outlook.com

***Abstract*—In the current state of the pandemic, local clinics have become more important than ever. We will present a glove with an embedded display which can rapidly detect Covid-19 related symptoms by touching the patient.**

***Index Terms*—TODO: add keywords in, this, format, for, each, key**

I. Introduction

In security, three main factors of authentication can be distinguished: *to know*, *to have* and *to be*. The first one requires the user to remember something, usually a password associated to the email or a username. The second one requires the user to own something during authentication, for example their own smartphone, or a physical key. The latter implies to recognize the user based on something that characterizes their uniqueness, for example from a fingerprint or their face.

Although passwords are inherently the weakest form of security to consider, they are actually the most widespread as form of user authentication on the web today [1]. Several are the techniques used to violate a password-protected account, among them there is phishing, guessing or spoofing [1]. Therefore, many have advocated for the use of a second factor for authenticating (2FA: 2 Factor Authentication) [2], thus limiting guessing and spoofing. Biometric authentication is becoming more and more widespread, especially with fingerprint scanners on phone, and in the last years with face scanner too. There are although several issues still not widely studied in biometric authentication, and systems still have defects that bring to their vulnerability. Bonneau et al. [2] surveyed these issues and also considered that some biological features have not been deeply studied yet.

Despite 2FA is extremely effective for better secure accounts, there are still attacks that can be carried out. For instance, OTPs can be victim of the so called real-time phishing attacks, which require the attacker to act as a "server-in-the-middle", pretending to be the real server and forwarding user's authentication requests and user's OTP to the real server [3]. However, message-based OTPs are the most used 2FA nowadays, especially because they rely on the assumption that a user will always carry their smartphone with them.

In this report, I will explain why security keys are theoretically better rather than other 2FA methods common today, bringing up problems in using other second factors and showing how they are avoided with a security key. I will then go through a sample web application that I developed in NodeJS[1] and VueJS[2], using Express[3] as framework, after looked at an overview of the cryptographic protocol used by Yubico[4] for their security keys.

II. Related Work

I would like now to report the most relevant related work in second factor authentication. Lang et al. [4] consulted a variety of excellent surveys work [2, 5, 6, 7] and listed five different technologies for 2FA, illustrating how security keys could fill some gaps in security left by those technologies. I will address four of them:

- **One-Time Passcodes:** Though OTPs provide more security than passwords, OTPs have a number of downsides. First, they are vulnerable to phishing and man-in-the-middle attacks, as I cited in Section I. Second, OTPs that are delivered by phones are subject to data and phone availability, while those that are generated by dongles cause the user to have one dongle per web site. Finally, OTPs provide a sub-optimal user experience as they often require the user to manually copy codes from one device to another. *Security Keys are resistant to phishing and man-in-the-middle by design; our preliminary study also shows that they provide a better user experience.*
- **Smartphones as Second Factor:** While leveraging the users phone as a cryptographic second factor is promising, it faces a number of challenges: for example, protecting application logic from malware is difficult on a general purpose computing platform. Moreover, a users phone may not always be reachable: the phone may not have a data connection or the battery may have run out.

[1] https://nodejs.org/en/
[2] https://vuejs.org/
[3] https://expressjs.com/
[4] https://yubico.com

Security keys require no batteries and usually have a dedicated tamper-proof secure element.

- **TLS Client Certificates:** Unfortunately, current implementations of TLS client certificates have a poor user experience. Typically, when web servers request that browsers generate a TLS client certificate, browsers display a dialog where the user must choose the certificate cipher and key lengtha cryptographic detail that is unfamiliar and confusing to most users. Accidentally choosing the wrong certificate will cause the users identity to leak across sites. TLS client certificates also suffer from a lack of portability: they are tough to move from one client platform to another. *Security Keys have none of these issues: they are designed to be simple to use, portable and fool-proof.*
- **Electronic National Identification Cards:** Some countries have deployed national electronic identification cards. Despite their rich capabilities, national identity cards require special hardware (a card reader) and thus are hard to deploy. Moreover, they are by definition controlled by one government, which may not be acceptable to businesses in another country and could arise a general concern about user's privacy. *Security Keys have no such downsides: they work with pre-installed drivers over commonly available physical media (USB, NFC, Bluetooth) and are not controlled or distributed by any single entity.*

III. Protocol Overview

References

[1] Joshua Reynolds, Trevor Smith, Ken Reese, Luke Dickinson, Scott Ruoti, and Kent Seamons. A tale of two studies: The best and worst of yubikey usability. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 872–888. IEEE, 2018.

[2] Joseph Bonneau, Cormac Herley, Paul C Van Oorschot, and Frank Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *2012 IEEE Symposium on Security and Privacy*, pages 553–567. IEEE, 2012.

[3] Katie Kleemola John Scott-Railton. Two-factor authentication phishing from iran, 2015. URL https://citizenlab.ca/2015/08/iran_two_factor_phishing/. Accessed: December 10, 2021.

[4] Juan Lang, Alexei Czeskis, Dirk Balfanz, Marius Schilder, and Sampath Srinivas. Security keys: Practical cryptographic second factors for the modern web. In *International Conference on Financial Cryptography and Data Security*, pages 422–440. Springer, 2016.

[5] Cormac Herley, Paul C Van Oorschot, and Andrew S Patrick. Passwords: If were so smart, why are we still using them? In *International Conference on Financial Cryptography and Data Security*, pages 230–237. Springer, 2009.

[6] Robert Biddle, Sonia Chiasson, and Paul C Van Oorschot. Graphical passwords: Learning from the first twelve years. *ACM Computing Surveys (CSUR)*, 44(4):1–41, 2012.

[7] Anil K Jain, Patrick Flynn, and Arun A Ross. *Handbook of biometrics*. Springer Science & Business Media, 2007.