ECE455 Week 2 Readings

Jonathan Lam

9/20/21

Contents

1 The Emperor's New Password Manager: Security Analysis of Web-based Password Managers

Li et al. (2014)

1.1 Overview

- "One primary, if not *the* primary, concern with password authentication is the cognitive burden of choosing secure, random passwords across all the sits that rely on password authentication" (1) – users have given up, choosing simple passwords and reusing them across many websites
- Benefits of password managers:
  - Automatically generate strong passwords
  - Automatically fill in passwords
  - Provide some protection against phishing attacks
  - Cloud-based synchronization allows for better usability
- Four key concerns:
  - Bookmarklet vulnerabilities
  - "Classic" web vulnerabilities
  - Logic vulnerabilities
  - UI vulnerabilities
- Diverse root causes:
  - Logic and authorization mistakes
  - Misunderstandings about the web security model
  - Vulnerabilities like CSRF and XSS
- All studied password managers rely on security through obscurity (proprietary, obfuscation/minifcation) and lack a published security architecture
- Systematic approach: identify the attack surface, security goals, and vulnerabilities
- Best mitigation strategy is defense in depth
- Ethics: all of the vulnerabilities were reported to the vendor in secret and fixed before the publication of this paper

1.2 Background

- Most password managers have a database of passwords and usernames, controlled by a master username/password
- Many password managers also provide automatic login (e.g., submitting a form) and require a "privileged browser extension or a bookmarklet" (3)
- Some password managers include the ability to share passwords with a collaborator; both users need accounts with the password manager
- Credentials are usually encrypted with a user-provided key (the *master key*). This can be derived from the master username/password.
- Bookmarklets can be used in lieu of browser extensions when the latter are not available, e.g., on mobile platforms. Bookmarklets use the existing bookmark functionality but invoke arbitrary Javascript code instead of going to a webpage; importantly, this code runs in the context of the current webpage (which could be malicious).
- Specific password managers
  - RoboForm: "Unless the user creates a master password to protect the files, these credential files are sent to RoboForm servers in the clear" (4)
  - NeedMyPassword also does not encrypt user passwords

1.3 Analysis

1.3.1 Threat model

- Main threat model is the *web attacker*: "controls web servers and DNS domains and can get a victim to visit domains controlled by the attacker" (5)
- Assume that the password manager is not malicious and does not steal sensitive data from web applications

1.3.2 Security goal

- Only one key security invariant: ensure that a stored password is accessed only by the authorized user(s) and the website the password is for

- Four security goals to achieve this invariant:
  - Integrity of master account
  - Credential database security (including confidentiality, integrity, and availability)
  - Collaborator integrity
  - Unlinkability: "the use of a password manager should not allow colluding web applications to track a single user across websites, possibly due to leaked identifiers" (5); "a password manager violates unlinkability if it allows tracking a user across web applications even in the absence of other techniques like web fingerprinting" (Bonneeau et al.)

1.3.3 Attack surface

- Web-based password managers store credentials in the cloud
- User logs into the service
- Access is through extensions, websites, or bookmarklets

1.4 Key vulnerabilities

1.4.1 Bookmarklets

- Bookmarklets cannot rely on any of the inbuilt JS API's, since they could have been overwritten with the website's malicious code
- Some newer browser extensions provide native or isolated APIs that help solve this problem, but browsers that don't support extensions don't have this capability
- Bookmarklet has to store the secret master key to decrypt the credential database
- This allows for a linkability attack
- Recommendation: run code in an iframe; same-origin policy enforces the integrity of the DOM APIs

1.4.2 Web vulnerabilities

- Writers of the password managers have to understand the basic web security model
- For example, "browsers share authentication tokens such as cookies across applications ... leading to attacks such as cross-site request forgery (CSRF)" (6)
- Case study: LastPass OTP
  - OTP must be able to authenticate to LastPass and allow the user to recover the master key; all without revealing anything extra (including the OTP itself) to LastPass servers
  - This attack gives attackers access to the LastPass account, including the encrypted database (for offline guessing), a DOS attack (deleting credentials), and breach of privacy (list of all accounts)
- Prevent storing secrets in JS files; web platform has weak security for script files (which can be imported from other domains); better to store secrets in HTML or JSON, which have better protections

1.4.3 Authorization vulnerabilities

- Sharing credentials increases the complexity of password managers
- "Confusing authentication for authorization is a classic security vulnerability" (6) "We separate out authorization vulnerabilities from web vulnerabilities since they are often due to a missing check at the server-side" (6)
- Recommendations: use a simpler model and don't use predictable identifiers

1.4.4 UI vulnerabilities

- Password managers are themselves susceptible to phishing
- Example attack is login to the password manager via an iframe; the iframe can be replaced by a fake website
- Iframes are less secure than opening in a new tab/window, because then the URL can be more easily verified by the user

2 Why no one uses encrypted email messages

- Two types of email encryption: encrypting own emails vs. encrypted email services
- In the latter, you're trusting the email service to securely handle your keys
  - In the past, the U.S. government has required Lavabit to give them access to the keys
  - I.e., latter is not safe
- Even with secure PGP emails, subject line, to, and from fields are generally sent unencrypted
- Need to understand how asymmetric encryption works, generate a key pair, and provide public key to others
- You have to remember a password to encrypt your private keys
- You have to agree on an email encryption standard
- Keep track of private key and revocation certificate to invalidate public key
- You need a mix of applications to be able to use it

3 I'm throwing in the towel on PGP, and I work in security

Valsorda (2016)

- Adoption issue: no one uses it, hard to use
- "But the real issues, I realized, are more subtle. I never felt confident in the security of my long-term keys. The more time passed, the more I would feel uneasy about any specific key ... A long-term key is as secure as the minimum common denominator of your security practices over its lifetime. It's the weak link" (2)
- Pets vs. cattle: pets are carefully maintained servers (and nursed back to health) whereas cattle are usually many replicated servers that can easily be replaced

- "Travel in particular is hostile to long-term keys"
- "I care much more about forward secrecy, deniability, and ephemerality than I do about ironclad trust" (2)

4 Questions:

- "... other password replacement schemes such as SSOs" (Li et al. 1) – what do SSOs have to offer
- In the LastPass bookmarklet vulnerability:
  - What are the purpose of the `r` and `rh` parameters? What do they represent?
- "The only secret in the bookmarklet code is an HMAC function (protected by DJS ...) that the password maanger iframe can use to provide *click authentication*" (Li et al. 12)
- Defensive Javascript (Li et al. 13)
- Web of Trust, fingerprints, or Trust on First Use (Valsorda 1)
- Secrecy, deniability, ephemerality, ironclad turst (Valsorda 2)
- Yubikey?