Security Enhancing Browser Extension

Team Information: Pratheek Reddy Amireddy (112073273), Mohith Kurakula (112504214)

PassMan

PassMan is a browser extension which helps users to browse more securely.

I. Features

The Features of the browser extension are:

1. Detect Password Reuse: PassMan detects whether the user on registering on a new website, is using a password that has already been used on a different website. The PassMan warns the user that the password he is using to sign up has already been used, and encourages him to use a different password.

2. Protect from Phishing Attacks: PassMan also detects whether the user is entering the correct password for the respective correct website. If PassMan detects any phishing attempts, it warns the user that he is using the password of a different website.

3. Protect from visiting unpopular websites: PassMan inspects all the links in all the webpages that the user has visited. If the user clicks on any of the links, PassMan checks if the link is in the Alexa top 10k website list. If the link is not in the list, then PassMan warns the user that he is about to visit a website outside of the top 10k website list. The user can choose to either visit the website once or whitelist the website. If the user whitelists the website, PassMan will not warn the user on visiting the website again. The user can also choose to not visit the website, in which case the user stays on the same page.

II. Architecture

There are three tasks that PassMan performs. The details for how each task is implemented is elaborated below:

Hashing of Password

SHA-256 is cryptographic hash function, it generates a unique fixed size 256 bit hash. We are using CryptoJS implementation of SHA-256.
Password Storage

Whenever user is signing up on a new website and when the user clicks on signup or register button, PassMan stores the hostname of the website and the hash of the password the user entered as a json object and then passes it to the background. We are utilizing CryptoJS's SHA256 hashing algorithm in order to hash our password. On the background script, PassMan stores the json object into an array and then stores the array in localStorage of Chrome Browser using the 'localStorage.setItem(key, stringify(userDetails)'. If Suppose, the user is using PassMan for the first time, then It creates a new array and then stores the json object into the array.

Detection of Password

PassMan detects the password entered by the user using jquery's keyup function. PassMan then detects whether the it is a signup form or login form using the .closest('form').find('button, input') function, which returns whether the form's button or input button is login or signup.

Signup Form

If it is a signup form, PassMan sends the hash of the password that the user is entering, to background script. In the background script, PassMan initially obtains the array of stored json object, with each object corresponding to a website, it obtains the hashed password using localStorage.getItem() function. PassMan then compares the hash of the password that the user has entered to the stored hash passwords. If any of the passwords match then PassMan sends a true flag back to the content script. PassMan then alerts the user that the password he has entered has already been used, If the user clicks on 'ok' then the password input field is cleared.

Login Form

If it is a login form, PassMan sends the hash of the password and also the hostname, to the background script. In the background script, PassMan obtains the hashed passwords using the localStorage.getItem() method. PassMan then checks whether the hash of the password that the user has entered to the stored hash functions. If the hashes match, then PassMan checks if the hostname of the corresponding hashed password and the hostname that the user is currently on. If the hostnames do not match, then there is a phishing attempt and thus the background script sends a true flag back to the content script. PassMan then alerts the user that the password he is entering belongs to a different website and thus if he clicks on ‘ok’, then the password field is reset to blank.

Modify-link clicking behavior

Initially when PassMan is loaded, the Alexa top 10k website list which is in .csv format, is automatically stored in the localStorage using setItem(key, arrayList), with each domain name an array element. The .csv file is read using the papaparse.js library.

Whenever, the user clicks on a link, PassMan obtains the link from the ‘<a>’ tag (i.e. the href attribute). It prevents the default of the link. It then obtains the hostname from the link and splits it and stores it in an array ‘host’, if the length of the host is greater than equal to 3 and for some special cases (i.e., .co.in, .net.cn. etc.), it obtains the last three elements in the array and concatenates them. If the above condition is not satisfied then it obtains the last two elements and concatenates them (this specific case if by keeping in mind that for some websites we get domain as en.wikipedia.org but our csv file contains only wikipedia.org). Thus, the domain name value is sent to the background script.

PassMan then compares the domain name which it receives in the background script with that of the stored domain names in our localStorage. If the received domain name is in the list of domain names stored, then a true flag is sent to the content script. PassMan then allows the user to visit the clicked link. If the received link is not present in the list of stored domain names, then it sends back a false flag. On receiving the false flag in the content script, the user is alerted that the link clicked is not in the top 10k Alexa list. He is given three choices: Don’t Visit, Visit Once, Add to Whitelist.

If the user clicks on Don’t Visit then user remains on the same page. If the user clicks on Visit once, then PassMan allows the user to visit the website only once, but next time on clicking the same link, the alert is shown. If the user clicks on Whitelist then PassMan adds the domain name to list of stored domain names. Again when the user clicks on the link, PassMan allows the user to visit the link since PassMan has added the domain name of the link to the stored list of domain names

III. Distribution of Load

The code for Detection of Password Reuse - login page and signup page, Storage of Password, Detection of Passwords on the wrong website has been done by Pratheek Reddy Amireddy.

The code for Modify-link clicking behavior has been done by Mohith Kurakula.

IV. Instructions to Set Up the Web Extension

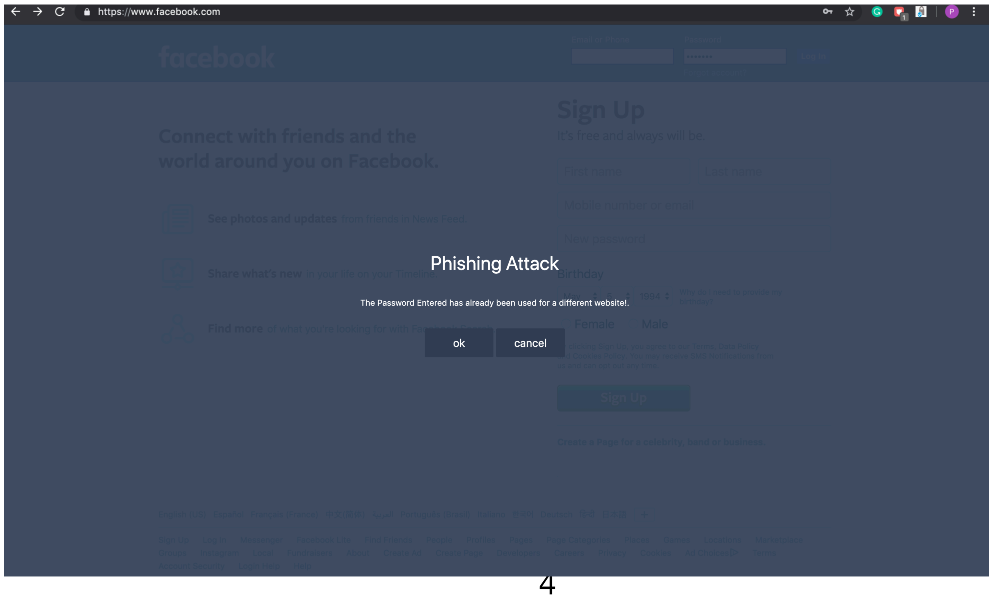
1. Follow the link: https://github.com/prateekami1994/Security-Enhancing-Browser-Extension-CSE-509
2. Clone the repository into your local computer
3. Goto Chrome web browser
4. In the omni search bar, enter the following: chrome://extensions
5. Toggle the developer mode to on
6. Click on Load Unpacked
7. Select the Browser Extension folder in the cloned repository
8. The extension should be displayed in the browser bar.

V. Usage and Test cases

Enter a password into the form's input password field and then click on signup. The browser extension stores the hash of the password that the user has entered.

Now try to login to a different website using the same password. The browser extensions shows an alert as follows:

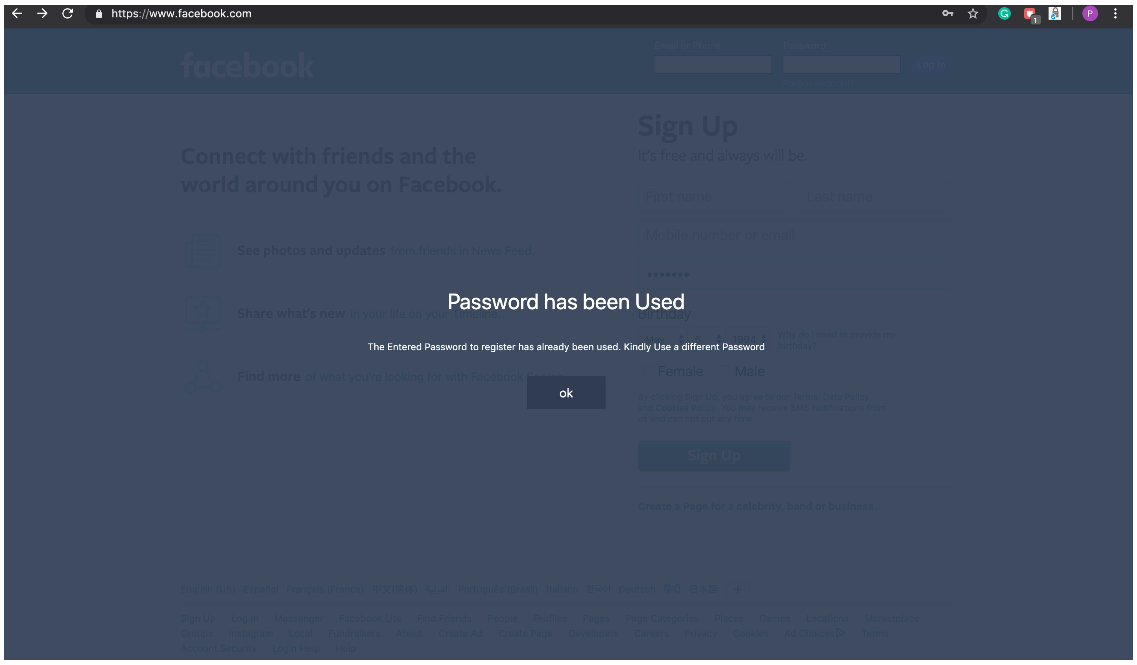
Initially, user has signed up on stack overflow. But on entering the same password on facebook



Phishing Attack

The Password Entered has already been used for a different website!.

If the user Signups on a different website using an already used password then the alert is as follows:

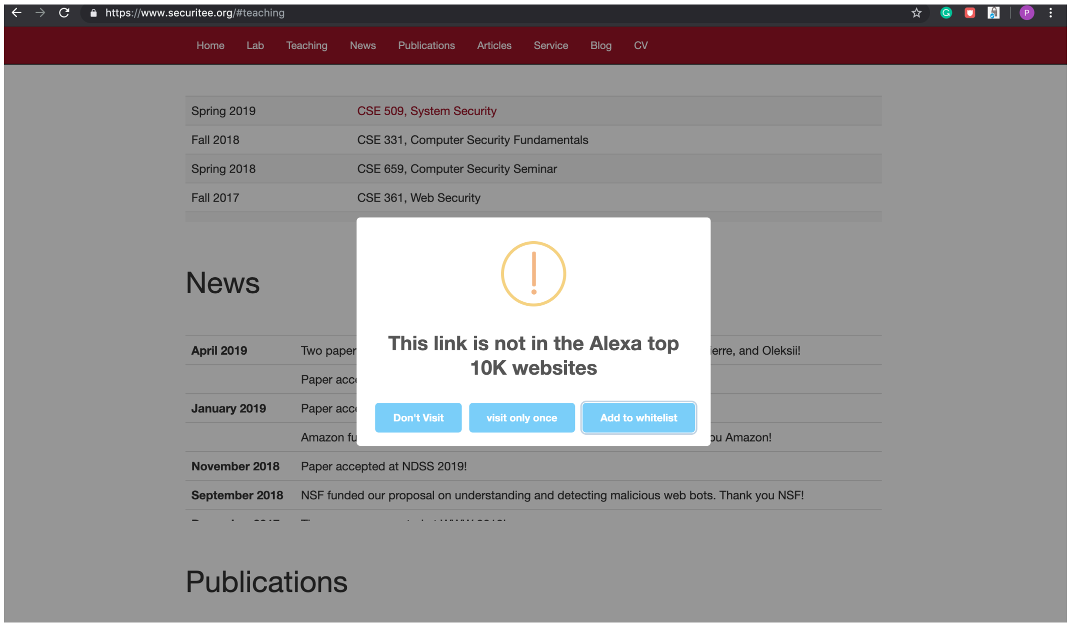

https://www.facebook.com

Password has been Used

The Entered Password to register has already been used. Kindly Use a different Password

ok

If the user visits a website that is outside of the Alexa top 10k website list then the following alert


https://www.securitee.org/#teaching

Spring 2019	CSE 509, System Security
Fall 2018	CSE 331, Computer Security Fundamentals
Spring 2018	CSE 659, Computer Security Seminar
Fall 2017	CSE 361, Web Security

News

This link is not in the Alexa top 10K websites

November 2018	Paper accepted at NDSS 2019!
September 2018	NSF funded our proposal on understanding and detecting malicious web bots. Thank you NSF!

Publications

VI. Third Party Content / Libraries

Cryptographic library: Crypto-js.js
Libraries for Alerts: sweetalert.js, jquery-confirm.min.js
Other Libraries: papaparse.min.js for parsing csv file, jquery.min.js for selecting actions and tags

VII. References

1. api.jqeury.com
2. developers.chrome.com/extensions
3. craftpip.github.io/jquery-confirm
4. www.papaparse.com