IJCSC 0973-7391 Volume 7 • Number 1 Sept 2015 - March 2016 pp.154-158 available online at www.csjournalss.com

Usability of Long Passwords in Web Application

Vipul Chudasama[1], Vishal Parikh[2] ,Tejas Chauhan[3]
[1][2]Assistant Professor ,Nirma University
[3]M.Tech Student, Nirma University
vipul.chudasama@nirmauni.ac.in, vishalparikh@nirmauni.ac.in,13mcei05@nirmauni.ac.in

Abstract: As the advancement and progression in internet technologies and computers, one of the main issues which came in the limelight is web security. Every person in the world is using the internet as their daily routine but they are not aware of the risks of it. People are doing an online transaction without knowing that if they don't use some security mechanisms, someone can steal their session ids and can empty their accounts. There are many types of authentication mechanisms available like passwords, biometric (voice, face and fingerprint) etc, but passwords are easy to implement and easy for users to understand. Passwords also have many drawbacks like if passwords are short or a dictionary word, it can be easy guessed or brute-forced by an attacker. It is a difficult task to create and remember passwords that are hard for an attacker to guess. Markov chains can be used to create password. So in this paper, we are also proposing a usable password generator which generates usable and memorable passwords for users.

Keywords: Web security, authentications, password, Markov chains.

I. INTRODUCTION

Web Security deals with the security of websites and web applications. It can be a website of a company with tells a user about them or a question/answer forum to help the programmer to solve their issues or a social networking site which a person uses to stay connected to his friends. Few years back most websites were static, but nowadays they became much more interactive. Users can see sliders, pop-ups and much more interactive content which makes a website more user-friendly. But as this user friendliness is developed by some client-side scripting languages, a new world of vulnerabilities also comes into the picture. Nowadays attackers are mostly using client-side scripting languages to create their exploits as it runs on client's machine, so they can fetch every detail related to the user like cookies, session ids, web browsing history etc.

A. Cross Site Scripting

Cross-Site Scripting is the most common vulnerability which exists in web applications and it occurs because of data that is entered by the user is not properly filtered. Most modern websites ask for user input for many purposes like feedback form, contact us form, search query etc. Web site owners think that user will write text in those input fields for which they are meant to be but attackers take advantage of that and write malicious codes in those input fields. If proper security mechanisms are not in place, then the attacker can execute those scripts on another user's machine as Javascript and other client side scripting languages runs in user's machine.[1]

B. SQL Injection

SQL Injection is an attack on web pages when user input contains some SQL statements. In this attack, some SQL queries are executed in the database results in theft of some confidential information and even database crash. For example, suppose we have a login form, which contains two input fields, one for username and another for a password. Now whenever the user enters a valid username/password combination, an SQL query is executed in the database to check that combination is valid or not[1]

Users generally create passwords which can be easily guessed or brute forced. To enforce users create secure passwords, websites started to use password policies and password strength meters. Unfortunately, password policies do not help to create secure passwords as users may fulfill policy requirements in a predictable way like, creating their new password based on an old password and also using the same password in multiple domains.

A passphrase is a password which contains a sequence of words. They are longer compared to ordinary passwords. Passphrases are more secure and easier to remember. Usable password generator automatically creates a passphrase from the input data supplied by the user. It is like a sentence which can be easily remembered but cannot be brute forced easily. We can also make it more secure by some extra policies of adding uppercase, lowercase, special symbols.

III. BACKGROUND AND RELATED WORK –USABLE PASSWORD GENERATOR

A. Usable Security

While designing a security system, we should consider the users which are going to use it and how they will use it. It should not be a frustrating part of users.

Measuring Usability

• Speed: How quickly can the task be accomplished?

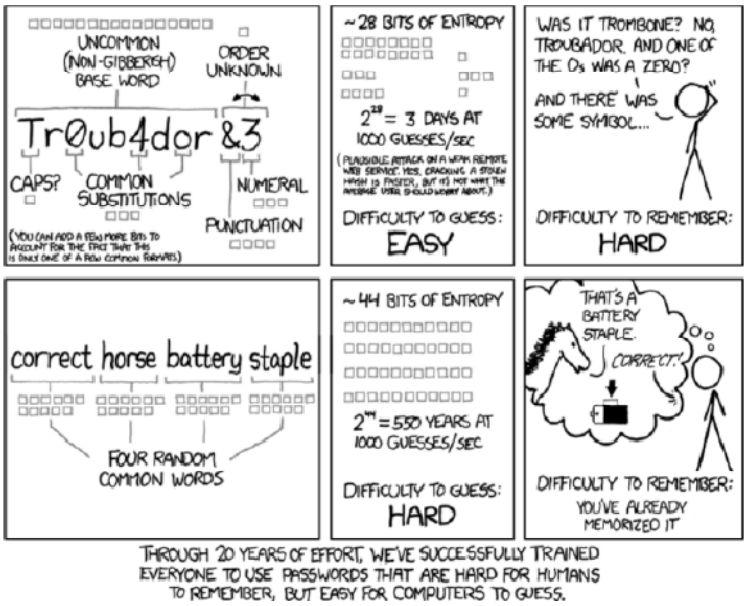
• Efficiency: How many mistakes are made in accomplishing the task?

• Learnability: How easy is it to learn to use the system?

• Memorability: Once learned, how easy is it to remember how to use the system? • User Preference: What do users like?

We searched for Mozilla Firefox add-on store and checked many currently available password generators. Mostly they were all with similar functionality. Whenever the user clicks on any predefined icon, a popup comes up and allows the user to configure some parameters according to which it will generate some passwords. Available password generators were able to create secure passwords but they were not memorable. User can't use those passwords as users can't remember it easily.

Here's an image [2] which perfectly differentiate between a secure password and a memorable password.



B. Password Entropy

Password entropy is defined as a password's randomness, in regards to how difficult it would be to crack. [3]

1. NIST Guidelines

NIST has analyze the information about human-generated password based on entropy [3]:

- The first character entropy is four bits;
- The next seven characters entropy is two bits per character;
- Entropy of ninth and through the twentieth character has 1.5 bits per character;
- Only one bit entropy is for 21st character.
- "Bonus" bits has been added for both to check dictionary based passwords.

2. Shannon's Formula

If we have a set of N symbols and if we select L number of symbols from that set then, the number of possible passwords can be calculated by raising N to power L. By incrementing either L or N we can more strengthen the generated password. A password's entropy H, can be given by following Shannon's formula

$$H = log_2 N^L = L \log_2 N = L \frac{\log N}{\log 2}$$

Where, N is the number of symbols and L is a number of symbols in generated password.

C. Markov Chains

Markov chains, named after Russian mathematician Andrey Markov is a stochastic process in which outcome of an experiment depends only on the outcome of the previous experiment, means the next state of the system depends only on the current state and not on other previous states. [5] A stochastic process is a mathematical model that evolves over time in a probabilistic manner. For example, if we model a baby's behavior as a markov chain we can consider eating, crying, sleeping and playing as different states. We can also list out all other possible states which will become our state space. Now a Markov chain tells us the probability (chances) of transition from one state to another e.g, the probability of a baby currently playing will sleep in next ten minutes without crying.

D. Comparison of Password Policies:

The eight conditions are detailed below [6].

• basic8survey: Password should be at least 8 characters long. It is in survey scenario.

• basic8: Password should be at least 8 characters long. It is in email scenario.

• basic16: Password should be at least 16 characters long.

• dictionary8: Password should be atleast 8 characters long. It should not have a dictionary word.

• comprehensive8: Password should be at least 8 characters long. It must have an uppercase and lowercase letter, a symbol, and a digit. It should not have a dictionary word.

• blacklistEasy: Password should be at least 8 characters long. It should not have a dictionary word. Here, the password is compared against simple Unix dictionary.

• blacklistMedium: Password should be at least 8 characters long. It should not have a dictionary word. Here, the password is compared against paid Openwall list.

• blacklistHard: Password should be at least 8 characters long. It should not have a dictionary word. Here, the password is compared against a five-billion-word dictionary.

Here is the graph[figure 1] which shows a number of guesses needed to crack a password which is created by different password policies listed above. Here it shows that basic16 policy which doesn't have any password restriction to compulsory use uppercase, lowercase and special symbol in the password and just due to its length, it is hard to crack.[7]

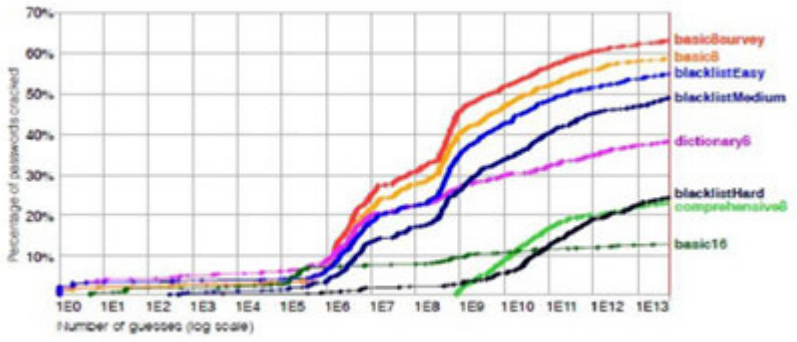


Figure 1. Password Policy

V. PROPOSED APPROACH AND IMPLEMENTATION – USABLE PASSWORD GENERATOR

We implemented a demo version of password generator which generates sentences as a password. It is implemented as a Firefox extension so anyone can install it in their browser. To create a Firefox extension, we need to get familiar with two technologies.

A. XUL

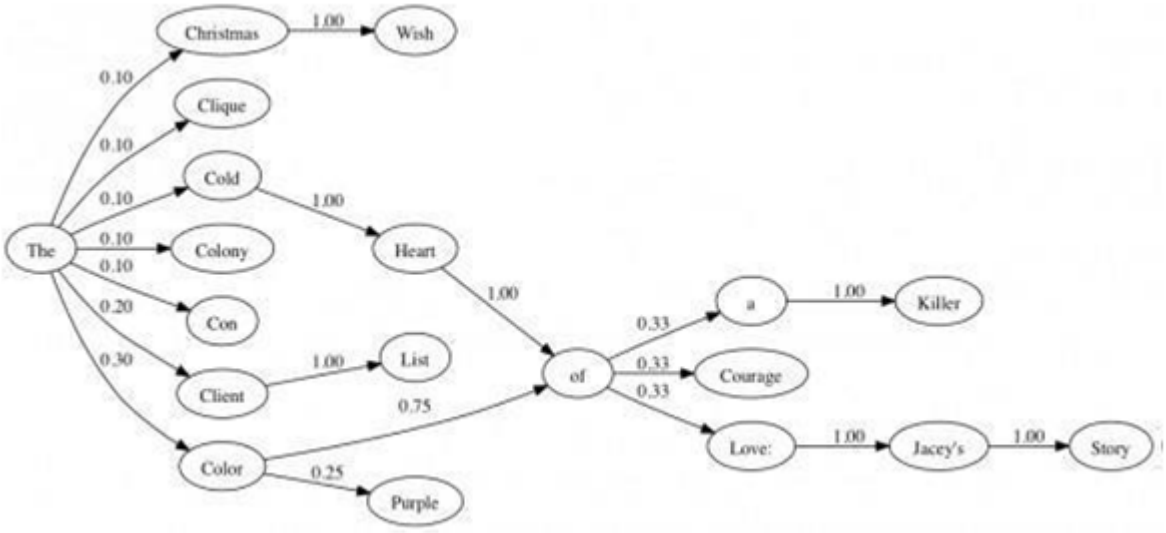
XUL (XML User Interface Language - pronounced "zool") is one of many technologies used for creating the user interface of Mozilla extensions. XUL is used to create portable and crossplatform user interfaces. It takes so much time to developan application for one platform, with XUL we can develop user interfaces which can be modified quickly and easily across multiple platforms.

B. Javascript

JavaScript is a client-side scripting language. It will run at client side (web browser) and used to develop interactive web pages. In Mozilla extensions, it is used to write application related functions.

In our demo implementation, we used a list of titles which is used to generate our Markov chain model. Now we can use that statistical information to generate new titles by selecting the first word at random and then selecting other subsequent words with the probability of how they are arranged in the original list. This gives us a list of different movie titles which are randomly created and will not be on that list.

First we list out all the unique words which appear in those movie titles. Next we calculated the probability of one word following the other word. In this way the whole graph is generated, a vertex represents unique words and value above each edge represent the probability of one word following the other word.

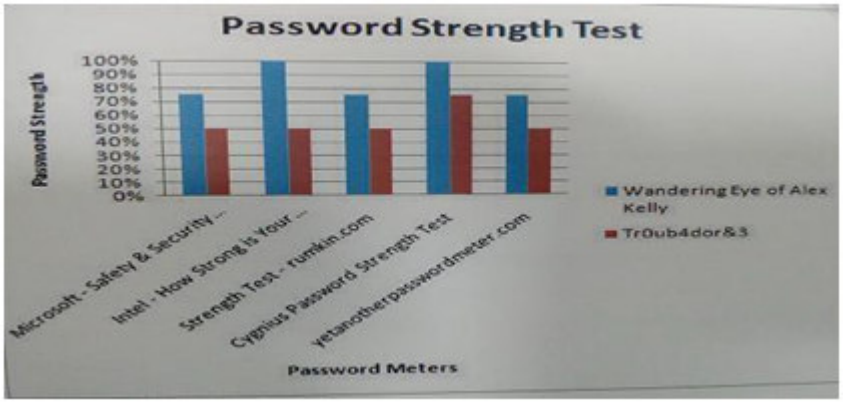


In above graph we can say the word “The” is followed by “Colony” with the probability of 0.1, while it is followed by “Client” and “Color” with the probability of 0.2 and 0.3 respectively. Now as we can see, we can traverse through any path of the graph and generate different unique titles.

VI. RESULTS AND FINDINGS

A. Password Strength Test

We used some well-known password meters to check strength of password created by usable password generator which is compared the with password created by a password generator which is available on firefoxaddon store. Here, "Wandering Eye of Alex Kelly" is the password generated by our password generator and "Tr0ub4dor3" is the password generated by other password generator available on add on store.



For Usable Password Generators, as per the different password conditions we tested, basic16 provides more security compared to other policies. Our demo implementation successfully generates passphrases which can be more memorable and hard for an attacker to break.

References

[1] Top 10 - 2013 (the ten most critical web application security risks (2013))," OWASP, 2013.

[2] https://www.outsideopen.com/password/

[3] "SP 800-63 - Electronic Authentication Guideline"). NIST. Retrieved April 20, 2014.

[4] U. W. Klaus Peter Jochum and B. Stoll, “Determining of reference values for nistsrm 610617 glasses following iso guidelines” 23rd USENIX Security Symposium, 2012.

[5] https://en.wikipedia.org/wiki/Markov_chain

[6] SarangaKomanduri , Richard Shay, Patrick Gage Kelley, , Michelle L. Mazurek, Blase Ur, Lujo Bauer, Nicolas Christin, Timothy Vidas, and Lorrie Faith Cranor, “Of Passwords and People: Measuring the effect of password composition policies” ACM, 2011.

[7] Patrick Gage Kelley, SarangaKomanduri, Michelle L. Mazurek,Richard Shay, Blase Ur, Timothy Vidas, Lujo Bauer, NicolasChristin and Lorrie Faith Cranor, “Guess again (and again and again) Measuring password strength by simulating password-cracking algorithms ” IEEE, Symposium on Security and Privacy 2012.