2024 Master's Thesis

Password Security Reinforcement via Combining Unicode Character Set

Supervisor: Prof. Tatsuya Mori
Research Guidance: Research on Networked Systems

A Thesis Submitted to the Department of Computer Science and Communications Engineering,
the Graduate School of Fundamental Science and Engineering of Waseda University
in Partial Fulfillment of the Requirements for the Degree of Master of Engineering
July 22nd, 2024

Student ID: 5122FG40

Xuhuai Nan

Abstract

Today's online passwords are mainly comprised of ASCII characters, even for users whose main languages are not English. This research firstly analyzes drawbacks of new authentication methods such as FIDO and stated the necessity of passwords. Meanwhile, real-world online password attack investigation and password preference investigation were done and its results indicated possible advantages of Unicode password. After that, this research proposed a potential way with a practical software to assist users with creating their passwords with non-ASCII (Unicode) characters and thus increase their password strength against attackers while maintaining ease of memorability. Another user survey about the tool proved its reliability and convenience, especially for no-English speaking users.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Background introduction

1.1.1 Authentication

As a cornerstone in security systems, the authentication typically facilitated through a combination of usernames and passwords, sometimes with additional factors. There are various existing authentication methods, including password authentication, SMS authentication and biometric authentication. Among them, the password authentication method still remains the most commonly used authentication method [1].

1.1.2 Password brute-force attack

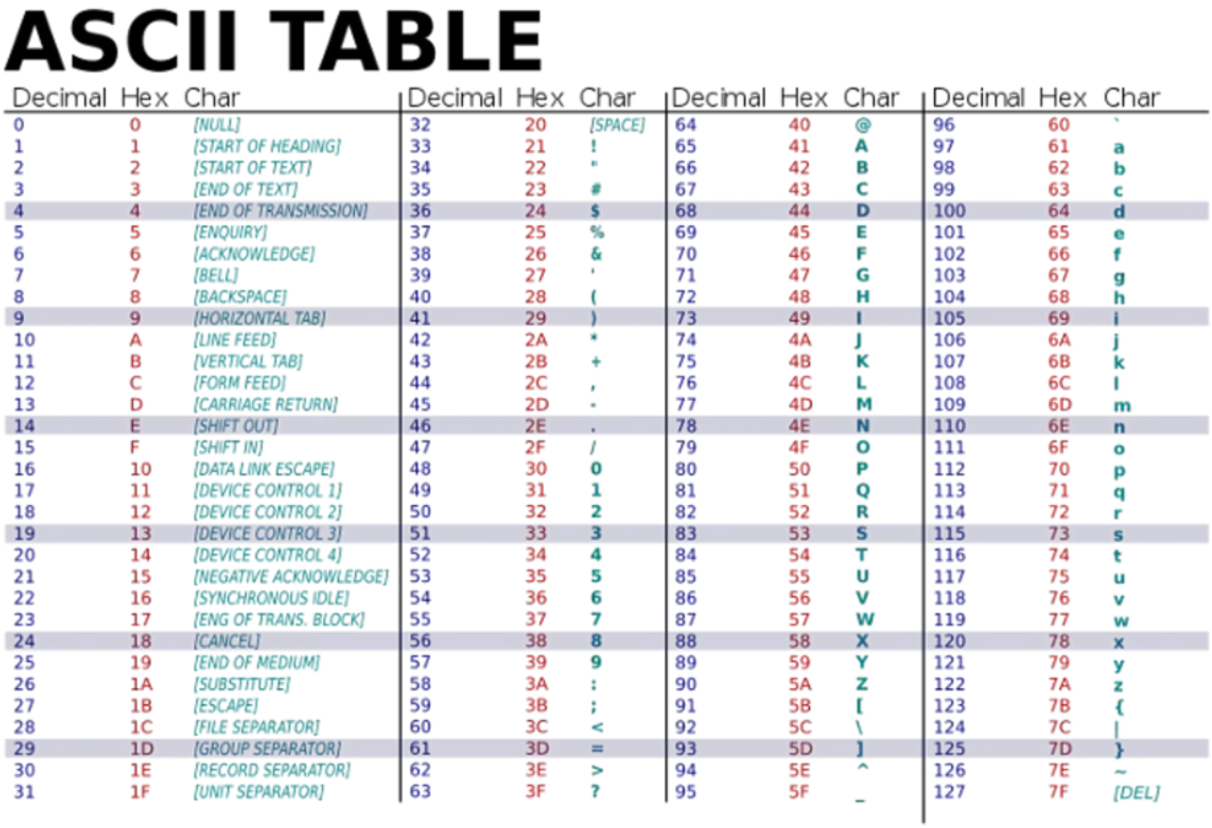
The password brute-force attack is the act of repeatedly attempting different passwords with the expectation of eventually achieving the success of login or decryption. It can be divided into online and offline password attack. The online password attack is mainly the act of testing passwords for online services provided by the Internet. Usually the server will limit the frequency of login attempts through accounts and source IP addresses for the purpose of users' identity security. The offline password attack does not rely on the Internet and the password is brute-forced locally. Attack targets include encrypted local files, encrypted USB drives and password hashes leaked from the server. To reduce the duration of brute-force attacks, multiple methods can be adopted, including using specialized, high-computing power processors, using common password dictionaries and using rainbow tables. Among these, users' hashed passwords leaked from web servers have become an important threat to end-user data security. According to posts from Have I Been Pwned, a large amount of data including password hashes is leaked on a near-monthly basis [2][3].

1.1.3 ASCII, Unicode and password entropy

ASCII, short for for American Standard Code for Information Interchange, is an encoding standard containing 128 characters (without consideration for Extended ASCII). This standard uses the lower 7 bits of each byte while retaining the eighth bit. This standard has been mature since 1967.

At the same time, the Unicode standard is a relatively new standard which is constantly being updated. As of now, the Unicode standard defines approximately 150,000 characters [4]. Among these characters are regional language scripts, symbols and the more recent emoticons. It can be inferred from the above that the ASCII set is only a small subset of the Unicode set.

The entropy of Unicode passwords is much higher than the entropy of ASCII passwords. Suppose the number of elements in a certain set is N and the length of a certain password is M, then the number of attempts required to crack the password is N to the M power(N^M). As the password length increases, since the base of the Unicode set is much larger than the ASCII set, the difficulty to crack Unicode passwords will be much higher than ASCII passwords. For example, assuming a



ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Fig. 1.1: ASCII Table

20-character password, theoretically, the number of attempts required to break a Unicode password is '150000^{20}', while it only takes '128^{20}' times to crack an ASCII password.

1.1.4 Character encoding standards and hash operations

Common character encoding standards include UTF-8, UTF-16, GB2312, etc. For each ASCII character that only occupies one byte, different character encoding standards do not affect their binary values. On the other hand, for non-ASCII characters such as Chinese characters, they have different binary values under different encoding standards. Different operating systems with different configurations may use different encoding standards. When a file created by one computer using a specific encoding standard get transferred to another computer, if the receiving computer uses a different encoding standard and cannot automatically detect the file's encoding, it may interpret the file incorrectly, leading to garbled text and other errors.

A hash function is a mathematical algorithm that converts input data into a fixed-size string of characters, commonly referred to as a hash value. With different input contents, it will always produce different hash values due to the function of its algorithm. It can determined whether multiple files hold the same content by comparing their hash values, which is mainly used for tamper-proofing, digital signatures and other occasions. For two non-ASCII strings with the same content which exhibit a visual uniformity, if their encoding formats are different, their underlying byte formats will be different, and thus their hash values will be different.

1.1.5 Password manager

Password managers can solve the tedious problems of password generation, management and input. Among them, notable password managers include Bitwarden, 1Password and LastPass. The basic usage process is that users register via email, set a master password, and then start setting passwords for each application and website, and synchronize information with the same account on their mobile phones and tablets. Users can customize the automatically generated password length and character set used.

1.2 Problem statement

1.2.1 Dilemma between password complexity and password memorability

In the digital age, where a significant portion of people' s lives is spent online, the importance of robust account security cannot be overstated. Passwords serve as the primary defense against unauthorized access to personal online accounts. In order to protect users' digital security, it is required by some service providers to create long passwords. However, long and complex passwords significantly compromise convenience and individuals frequently encounter difficulty in recalling them. It is a dilemma between creating long and complex passwords and remaining memorability. Unfortunately, this dilemma could prompt individuals to resort to bad strategies that may compromise either security or memorability, sometimes both.

1.2.2 Limited character sets for password creation and input

In the realm of cybersecurity, the importance of robust password creation cannot be overstated. However, one often overlooked challenge is the limitation imposed by character sets. No-English speaking users may have noticed that when it comes to create or enter their passwords to websites or applications, their input method automatically becomes ASCII character based input method, which means they cannot create or enter their passwords with characters based on their own language easily. While ASCII has served as the foundation for text-based communication in computing for decades, its constrained character set, resulting reduced password entropy, poses challenges in the context of password creation and input.

1.3 Outline of thesis

The thesis structure is outlined below:

Chapter 1 serves as the foundational introduction to the research topic. The first part is background introduction. It explains several concepts related to password. The second part states the problems related to password authentication nowadays.

Chapter 2 delves into existing research and developments related to the topic. The first related research is about the usage of the emoji password. This section provides an overview of emoji passwords, detailing their benefits and limitations. The second related research is the passwordless solution: FIDO Passwordless Authentication. This section introduces the FIDO Alliance's initiatives aimed at reducing reliance on passwords. In addition, it explains FIDO2 passwordless authentication and its security benefits as well as limitations.

Chapter 3 outlines the proposed solution and its implementation. The main idea of the thesis

is proposed: proposal for a client-side assistance tool to incorporate Unicode characters, which are largely based on languages, into passwords. It introduces the tool' s creation process and its demonstration afterwards.

Chapter 4 details the experiments and investigations conducted. It includes real-world password attack experiment, user password preference investigation and the usage feedback investigation of the tool created by this research.

Chapter 5 wraps up the research, highlighting conclusions and future directions. Firstly, it summarizes and concludes chapters above. Secondly, it states future works to be done.

Chapter 2 Related works

Numerous ongoing research efforts and innovative projects are actively dedicated to enhancing users' identity security on the Internet, addressing a lot of challenges and vulnerabilities inherent in the digital landscape. These endeavors encompass a diverse array of approaches, methodologies, and technologies, each aimed at fortifying the integrity of individuals' online identities and safeguarding against a wide variety of cyber threats and malicious activities.

2.1 User authentication enhancement with Emoji passwords

Emoji password is the password that uses emoticon characters instead of normal ASCII characters. There is a conceptual design to achieve user authentication by using emoji passwords instead of ordinary numeric passwords [5]. Researchers have a profound conclusion that the emoji password improves security, while their visually distinctive nature aids users in recalling their passwords more effortlessly [5].

However, despite the emoji password' s innovative approach and potential benefits, several significant drawbacks must be considered when evaluating their practical implementation.

Long and inconsistent length. While all of the Unicode values of Emojis are very large, they differ greatly in value and length. The Unicode value of "smile emoji" introduced in 2012 is ('\U1f600'). However, new emojis are added Unicode character set periodically. Those new emojis have much longer length than originally added emojis. The Unicode value of "fruit emoji", introduced in 2023, is (\U1F34B200D1F7E9). In this case, if the password input field is not specially processed, "smile emoji" represents ** and "fruit emoji" represents ***** in the password input field. Therefore, users may be confused when using Emoji passwords.

Fig. 2.1: The smile emoji

Fig. 2.2: The fruit emoji

Relatively bad cross platform compatibility. Different operating systems may render the same emoji in a variety of distinct ways. The emoji password does not have good compatibility [6]. For example, an emoji that looks a certain way on an iOS device may appear quite different when

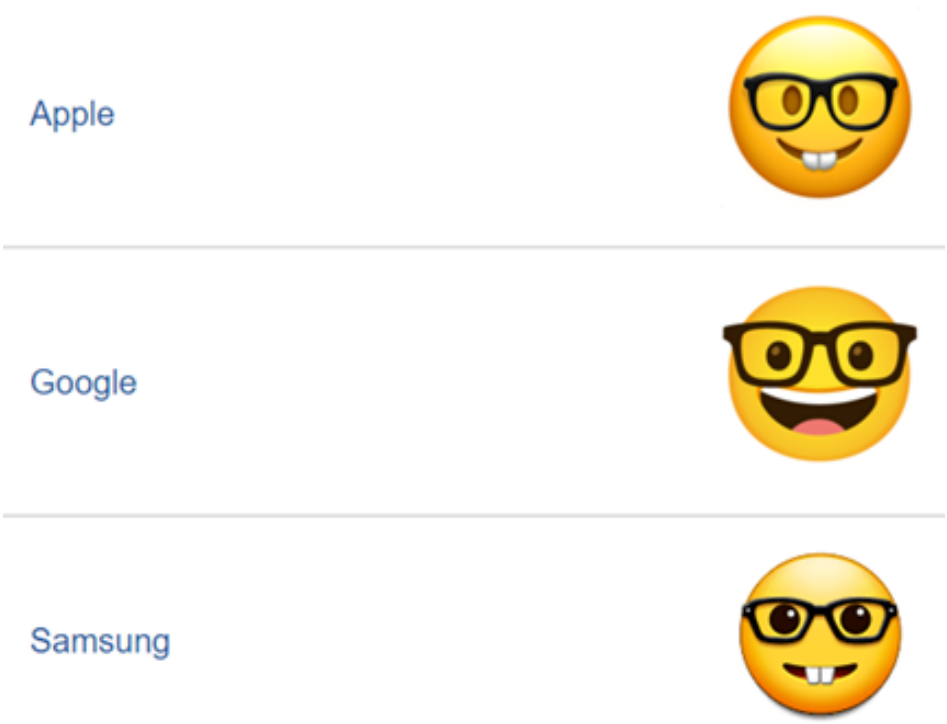


Fig. 2.3: Renderings of the same emoji 'nerd face(U+1F913)' in different platforms

Neaural faces	Smiling faces	Hands
😐	😃	✋
😑	😀	🤚

Similar emojis

Fig. 2.4: Similar emojis

viewed on an Android device, a Windows computer, or other systems. This inconsistency in rendering results in the emoji lacking good compatibility, as users across different platforms might experience varied interpretations and visual representations of the same symbol, which can lead to confusion and potential login failures. For example, the emoji designated as "nerd face (U+1F913)" is rendered differently across various platforms, as illustrated in Figure 2.3.

Emoji similarity. Another study on emoji passwords also found that users had difficulty distinguishing highly similar emoji passwords [7]. Selecting the correct emoji can be more challenging due to their visual similarity. Users might accidentally choose an emoji that looks almost identical to the one they intended to select, resulting in incorrect password entries. Examples of similar emojis are shown in Fig 2.4.

2.2 Passwordless solution: FIDO passwordless authentication

The FIDO Alliance is an alliance dedicated to establishing new authentication methods and reducing the world's overreliance on passwords [8]. According to the white paper published by FIDO [9], the solution currently proposed by FIDO is to avoid the usage of passwords through WebAuthn.

Specifically, it relies on the public and private key verification technology. The server uses the public key to send a challenge to the client, and the client uses a corresponding private key to sign (encrypt) the challenge and send it back to the server, which is then verified by the server. For the client, the private key can be saved in an external USB device or the electronic device itself (such as a mobile phone and a computer).

Also, from 2021, companies like Apple and Google introduced passkeys, a wrapper of the current FIDO2 approach, while they add additional features like syncing passkeys across users' devices. Popular media consider passkeys as the replacement of current passwords [10][11]. Moreover, researchers also found FIDO2 passwordless authentication more secure than password-based authentication [12].

Nevertheless, just like emoji passwords mentioned above, it is essential to take several major drawbacks into consideration when assessing its practical implementation.

High adoption barriers. Firstly, the adoption of FIDO2 is not so fast as expected. Companies have concerns such as costs and too big changes related to deploying FIDO2, according to a research [13]. While FIDO is a standardized protocol, variations in implementation and compatibility across different platforms and services can create fragmentation issues. The research also states that passwords will not be got rid of in the near futures [13]. Besides, an article considered it very hard to kill the online password by inferring from Apple's new password app [14].

Recovery issues. While password recovery processes are well-established, the recovering process of FIDO can be more complex as FIDO protocols rely heavily on a single device for authentication. If a user loses their authentication device or biometric data changes, the recovery process may be cumbersome. It creates a significant barrier to account recovery, as the private key used for authentication is stored exclusively on the device [15].

Chapter 3 Proposed approach

3.1 Main idea

In order to protect users' accounts security, online service providers could adopt multiple server-side strategies such as implementing MFA (Multi-factor authentication) and using more complex password hashing algorithms. However, not all of them can achieve this and certain online service providers exhibit a concerning lack of diligence in safeguarding users' account security.

While those server-side measures could be transparent and invisible to users, it is users' rights and obligations to proactively take client-side measures to safeguard their account security. This paper advocates for the use of Unicode passwords as an effective client-side strategy.

More than just world-wide emojis and math symbols, the Unicode character set also contains many language-based subsets. People who speak a No-English language have a much higher usage rate of their language-based Unicode subset than merely ASCII character set on the Internet. If people use their own language-based Unicode subset as the password character set, their account may be more secure. A thorough analysis of the detailed information is provided in Chapter 4.

However, due to the limitation of the password field, it is not convenient to directly input Unicode characters into the password field. As a result, they are almost forced to use ASCII set based passwords.

To effectively address the issue of incorporating and inputting Unicode characters into the password field, a comprehensive solution requires the development of a client-side Unicode character input helper tool. This tool is actually a web extension, tailored to cater to the diverse needs and preferences of users.

3.2 The tool's creation process

The browser extension is created based on an open-source browser extension named Bitwarden [16]. Additional functions related to Unicode password are added. The first function is implemented by creating a new content script file with a special input listener and re-configuring the manifest file. The second function's implementation involves creating and modifying several existing TypeScript components related to the mechanisms of password creation and generation within the extension.

3.3 Functionalities and demonstration of the tool

Password filling assistance. The first and the most important function of the tool is to assist users smoothly in inputting Unicode characters into password fields in websites' login page. With the developed tool, additional troublesome steps such as "copy and paste" are no longer required. Instead, Unicode passwords can be just typed directly and normally, as shown in Fig. 3.1 and Fig. 3.2.

Fig. 3.1: Demonstration of the first function in Chinese

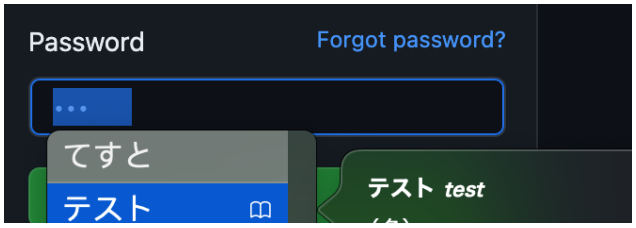


Fig. 3.2: Demonstration of the first function in Japanese

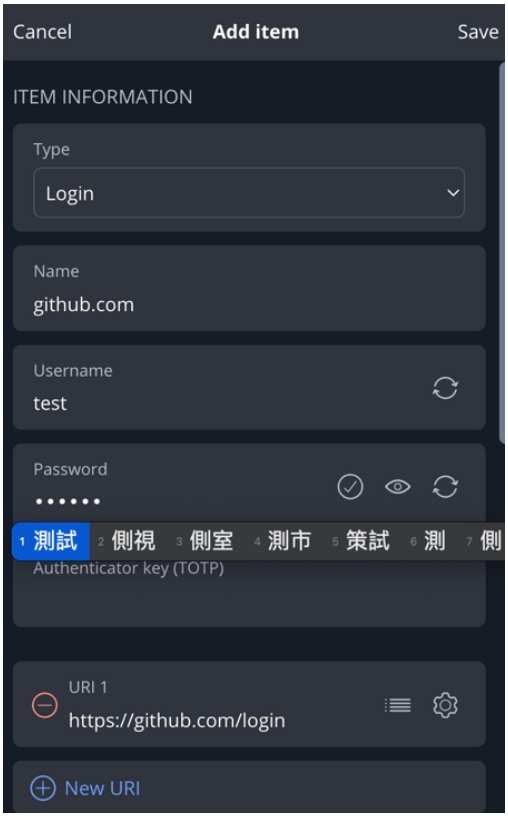


Fig. 3.3: Demonstration of the second function

Password generation and query. The second function is to help users smoothly create and save Unicode passwords inside the browser extension. It supports the generation of passwords of a specific length based on Chinese, Japanese, ASCII characters, numbers and special characters, as shown in Fig. 3.3.

Chapter 4 Experiments and investigations

4.1 Real-world password attack behavior investigation and observation

4.1.1 Experiment introduction

In order to collect which passwords are frequently used by real-world attackers nowadays, an experiment was conducted on several honeypot servers exposed to the public accessible Internet. Those servers had two service ports opened. One port was port 22(SSH) and the other was port 80(HTTP). Credentials for the SSH service were recorded using Pluggable Authentication Modules (PAM). For the HTTP service, credentials were logged through a PHP script. Additional information, such as the source IP address of the request and the corresponding timestamps, was also recorded in a log file for each service. The duration of this experiment was one month.

4.1.2 Analysis of Experiment results

One month after the experiment started, there were around 122 thousand network requests collected in log files.

The majority of login attempts were periodically carried out by malicious bots. Robot-like features can be discovered by analysing network requests' IP addresses and their corresponding timestamps in the log file. For instance, by filtering the log file with one certain IP address, the time differences between each neighboring timestamps were exactly 60 seconds, which was easy for bots but hard for human to achieve. In addition, each recorded IP address was checked against an online IP reputation service[17] and statistical results showed around 96.3% of them had a serious reputation issue. By combining the above two factors, it can be determined that most login attempts were conducted by malicious bots periodically.

All of the passwords in log files were comprised of ASCII characters. By comparing collected passwords with common password dictionaries[18], results showed that most of the collected passwords were common passwords such as '123456', 'password' and 'qwerty'. No password that consists of Unicode character was found in the log file at the time being. Table 4.1 shows distribution of common passwords among collected password data.

In conclusion, online password attacks are prevalent on the public Internet. These attacks typically occur silently by bots and are difficult to detect. Passwords used in these attacks inside this experiment are all ASCII-based passwords. Therefore, utilizing Unicode passwords may enhance account security against such online password attacks.

Table 4.1: Distribution of Common Passwords Among Collected Password Data

Password Category	Percentage of Total
Top 100 passwords	63.2%
Top 1000 passwords	78.6%
Top 10000 passwords	92.3%
ASCII passwords	100%

4.1.3 Analysis of possible reasons behind the experiment's results

Relatively poor Unicode support for password dictionary creating tools, password cracking tools and password dictionaries. One of password dictionary creating tools is named "hashcat" [19], which does not support creating Unicode password dictionaries. With the help of specific rules, it can create special and customized password dictionary based on existing dictionary. For example, rule "$a" means append character "a" to the end of each existing word. However, when "a" become any Unicode character, it would fail to create customized password dictionary along with the error message "Skipping invalid or unsupported rule", as shown in Fig. 4.1.

```
C:\tmp> cat origin.txt
password
123123
123456

C:\tmp> cat ascii.rule
$a

C:\tmp> hashcat -r ascii.rule --stdout origin.txt
passworda
123123a
123456a

C:\tmp> cat unicode.rule
$名

C:\tmp> hashcat -r unicode.rule --stdout origin.txt
Skipping invalid or unsupported rule in file unicode.rule on line 1: $名
No valid rules left.
```

Fig. 4.1: ASCII rule vs Unicode rule on customized password dictionary creation with the help of hashcat

Another tool is "thc-hydra" [20]. Similar to "hashcat", its "-x" option, which specifies the character set used to crack passwords, exclusively accept ASCII characters and is unable to process any Unicode character.

For popular password dictionaries such as rockyou.txt [21], a standard file in password cracking

which originated from a significant data breach, the majority of its passwords are in ASCII format, with only few being in Unicode format.

Due to the limitations above, it is more difficult and troublesome for a regular attacker to attack password comprised of Unicode character instead of ASCII character. That could be the reason why all of the passwords harvested from the experiment above were ASCII passwords.

4.2 The user study of login method and password preference

4.2.1 The initial user study introduction

A multinational user study was undertaken to examine user preferences regarding login methods and password selection across diverse linguistic backgrounds. The primary objective of this study is to figure out the prevalence of password usage and the current status of Unicode password adoption.

The survey was comprised of three languages: Japanese, Chinese and English. The Japanese version survey was conducted in a platform called "Freeasy". The Chinese one was conducted in a platform called "wenjuanxing" and the English one was conducted in a platform called "Qualtrics". In each platform, above 100 users were questioned about their basic information as well as questions related to password creation and preference.

4.2.2 The initial user study result: participants' basic information

The participants' first language distribution includes 107 Chinese speakers, 100 Japanese speakers, and 123 English speakers. Regarding education levels, 91 participants have a graduate degree or higher, 173 are undergraduates, 58 have a high school education, and 8 have a middle school education or lower.

Table 4.2: Participants' first language distribution

First language	Number of participants
Chinese	107
Japanese	100
English	123

Table 4.3: Participants' education level distribution

Education level	Number of participants
graduate or higher	91
undergraduate	173
high school	58
middle school or lower	8

4.2.3 The initial user study result analysis: login method preference

Among various login methods shown in Fig 4.2, Fig 4.3 and Fig 4.4, "password" stands out as the predominant choice for users of both English and Japanese languages, while maintaining its status as the second most utilized login method among Chinese users. On the whole, old-school "password" still remains its popularity nowadays.

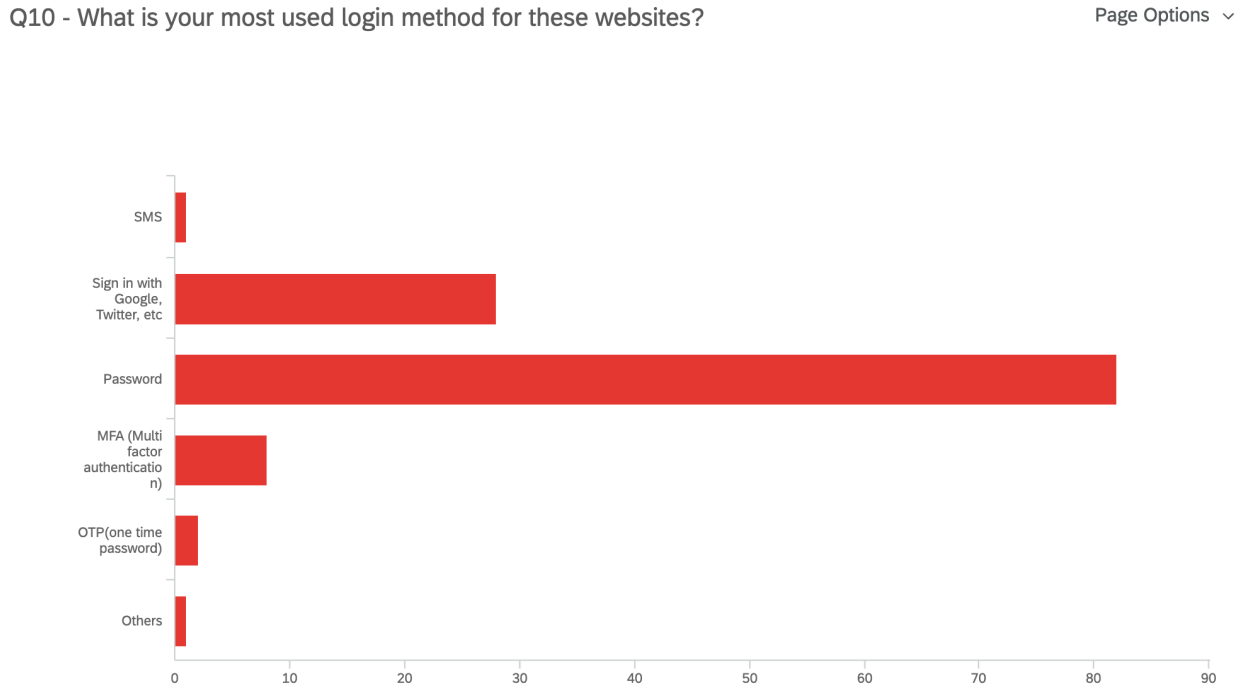


Fig. 4.2: Login method preference of the English version survey

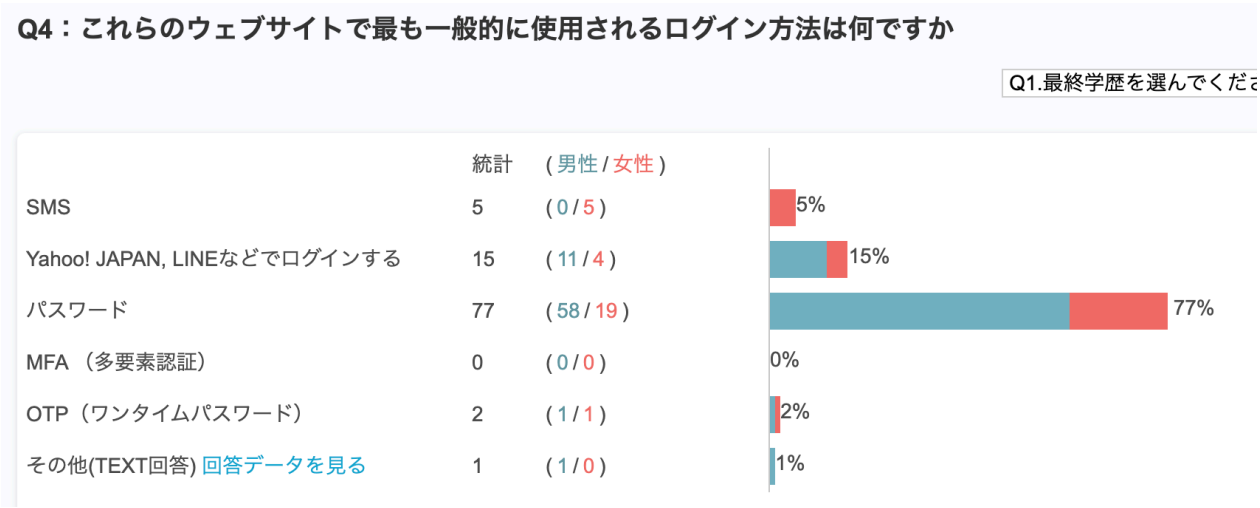


Fig. 4.3: Login method preference of the Japanese version survey

在这些网站中，您最常用的登录方式是什么？ [单选题]

选项	小计	比例
短信	17	15.89%
使用微信，QQ等登录	42	39.25%
密码	26	24.3%
多重身份验证（MFA）	20	18.69%
一次性密码（OTP）	1	0.93%
其他 [详细]	1	0.93%

Fig. 4.4: Login method preference of the Chinese version survey

4.2.4 The initial user study result analysis: password character usage status

In general terms, most respondents do not have Unicode character in their password. As the education level goes higher, larger proportion of people have Unicode character in their password. It can be inferred that "Unicode password" exhibits significant potential for utilization.

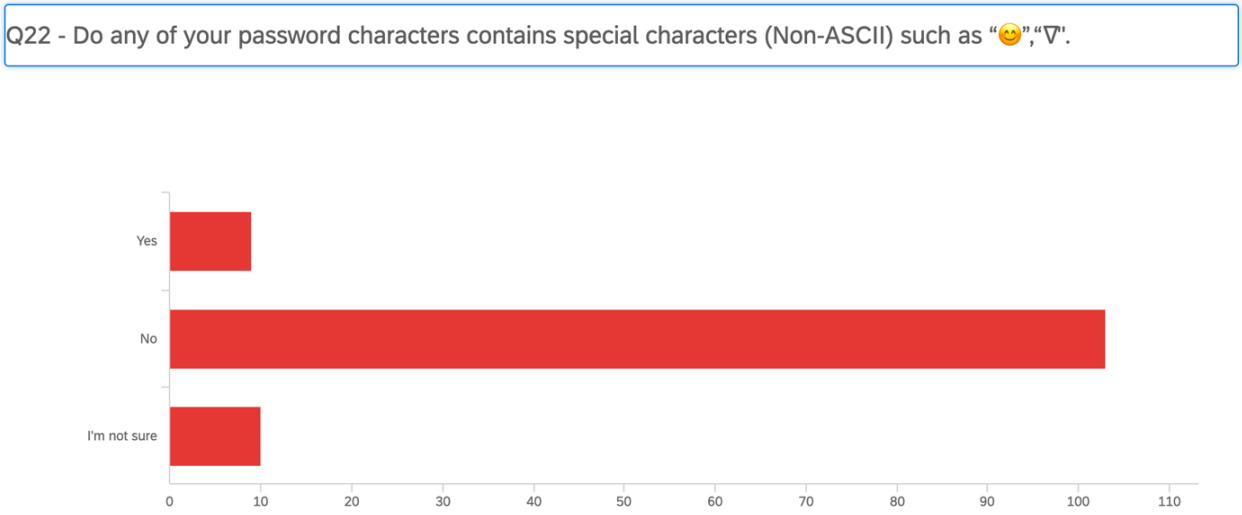


Fig. 4.5: Whether participants' passwords contain Unicode(non-ASCII) character

4.2.5 The initial user study result analysis: password transliteration preference

For no-English speaking users, password transliteration means transliterate characters in their own language into the ASCII character set. For example, "nihao" for "好" in Chinese and "neko"

for "猫" in Japanese. According to the survey conducted, roughly 28% Chinese users and 23% Japanese users have preference for password transliteration. Among them, most of the users prefer characters which constitute their own language to create non-ASCII (Unicode) password. If those people can easily type Unicode password, they may quickly adopt their own frequently used Unicode characters based on ASCII characters from password transliteration.

4.2.6 The initial user study: conclusion and summary

The study revealed that "password" still remains the most preferred login method in general. "Unicode password" exhibits significant potential for utilization and has a promising future. The Unicode password adoption may be easier for those participants whose first language was not English, because they had a password transliteration preference in their ASCII password.

4.3 User feedbacks associated with the tool created in Chapter 3

4.3.1 The follow-up user study introduction

Another cross-region user study was undertaken to collect the Unicode password input helper tool's feedbacks. The selected participants in this user study had either experiences or preferences of using Unicode characters as their passwords.

All participants had been notified of the survey's privacy policy, and each one had formally agreed to its terms and conditions. They were requested to download, utilize and provide feedbacks about with the Unicode password tool.

To be specific, the Unicode password web extension was provided from a Google Drive link along with a README.md file containing the detailed instruction about how to install it: open Google Chrome or Chromium-based browser, go to the "chrome://extensions/" URL, turn on the "developer mode" toggle switch and finally click the "load unpacked" button to load the extension. The usage tutorial, which is mentioned in Chapter 3, was also provided and explained to them.

Task One: passwords creation and recall. Participants were asked to create both ten ASCII passwords and ten Unicode passwords through the utilization of the tool as shown in the figure 3.2, while they don't really need to create accounts or modify passwords for real-world websites. Additionally, we advised the participants to refrain from using their actual passwords and instead generate passwords that mimic the type they might typically use. These data would be saved in the Unicode password input helper tool.

For the second part of Task One, participants were instructed to recall the passwords they had previously created after one week, two weeks and three weeks. Participants had two options to determine how many passwords they could recall after some time. The first option was to recall passwords and compare them with previously created passwords saved in the tool. Participants just needed to report how many passwords they could remember to the survey's investigator. The second option was that participants were encouraged to share passwords they created to the survey creators. They needed to submit passwords they created and recalled to the survey's investigator. The survey's investigator compared those passwords and determined the final password recall ratio.

Task Two: usability experiment. Participants were asked to use the tool across at least five of their most frequently visited websites by trying to create Unicode passwords and modify existing ASCII passwords to Unicode passwords. They were invited to highlight the strengths of the tool and explain what they appreciate about it. Meanwhile, if any bug was found, participants could

report them in the corresponding survey, detailing the issues encountered and any steps that might help in reproducing the errors.

In addition to two practical tasks above, participants were encouraged explore and utilize the tool according to their individual preferences and inclinations.

Task Three: provide feedback via surveys. The participants' feedbacks were obtained through a corresponding usage feedback survey. The first part of questions is related to their basic information such as the first language and education level. The second part of questions is relevant to the task completion status and user experience about the tools mentioned above. The analysis of the user study's result is in the following parts.

In order to be as close to reality as possible, it was encouraged to use personal accounts. For users who were concerned about privacy and potential errors which may be caused to their personal accounts by using Unicode password, they could use emails provided by the experimenter to create accounts or change passwords.

4.3.2 The follow-up user study result: participants' basic information

This user study was a combination of offline and online with a total number of 41 participants, among which 39 participants' first languages were not English. In table 4.4, the majority of participants speak Chinese (19), followed by Japanese (18) and English (4). In table 4.5, most participants have an undergraduate education (19), with 13 having graduate or higher education, 8 having a high school education, and 1 having a middle school or lower education. These distributions highlight the linguistic and educational distributions among participants.

Table 4.4: Participants' first language distribution

First language	Number of participants
Chinese	19
Japanese	18
English	4

Table 4.5: Participants' education level distribution

Education level	Number of participants
graduate or higher	13
undergraduate	19
high school	8
middle school or lower	1

4.3.3 The follow-up user study result: the analysis of created passwords

By analysing Unicode passwords shared by participants, Japanese and Chinese participants were more likely to use characters in their own language as Unicode characters while English participants tended to use variants of alphabet such as $\hat{a}$.

When comparing passwords collected by participants with password in several well-known password dictionaries, we found that collected Unicode passwords had zero hit rate in all three dictionaries. On the other hand, a certain amount of ASCII passwords created by them hit in some password dictionaries. Detailed information is in Table 4.6. Online passwords are usually stored as hash format inside service providers' databases. Even if there is a online credential data leak, it is not so easy and convenient for hackers to crack Unicode password with popular password dictionaries. They have to build a customized dictionary or crack hashed passwords by incremental bytes, which may be unusual and troublesome for them.

Table 4.6: Percentage of collected passwords contained in password dictionaries

Password dictionary	**Hit rate: collected ASCII passwords(%)**	**Hit rate: collected Unicode passwords(%)**
rockyou.txt [21]	26.5	0
john.password.lst [22]	23.2	0
top-100000.txt [23]	18.1	0

Password entropy is a measure of how unpredictable a password is, often calculated in bits. The formula for password entropy is as follows:

$$H = L \cdot \log_2(N)$$

- H is the entropy in bits.
- N is the number of possible symbols (character set size).
- L is the length of the password.

The average password entropy of collected ASCII passwords was 53 while that of collected Unicode passwords was 129.

Results in Table 4.7 from Task One which asked participants to recall passwords they created after some time showed that participants remember Unicode passwords slightly better than ASCII passwords that create.

Table 4.7: number of recalled passwords in different times

Time elapsed	**Average ASCII passwords recall count**	**Average Unicode passwords recall count**
one week	5.3	5.5
two weeks	4.1	4.3
three weeks	2.5	3.2

From the analyses above, it can be inferred that Unicode passwords can increase password strength while remain password memorability relatively unchanged.

4.3.4 The follow-up user study result: traditional ways to input Unicode password

Participants were inquired about their personal experiences and opinions concerning their own ways to input Unicode password. As their responses indicated, there were two primary methods they intended to utilize.

Copy and paste. A user can type Unicode characters in certain locations such as the notepad app. Then he can copy the content and paste it into the password field. It is considered an easy and direct method to input Unicode password by most participants. Nevertheless, there exists several disadvantages in accordance with some tech-savvy participants' responses. In one participant's opinion, he was worried that certain unwanted software programs, which periodically read the device's pasteboard's content, may acquire his password. If he forgets to clear the pasteboard's content after "copy and paste", the password may remain permanently on the pasteboard, resulting in a more significant security risk.

Key combination or special keyboard. With certain key combination such as "alt+a = å" or a language specific virtual keyboard, Unicode characters can be created and inputted into the password field. Fig. 4.6 shows what does the keyboard looks like when holding both "shift" and "option" key on macOS. When asked if any drawbacks exist, some surveyed participants answered "yes". Firstly they mentioned that the "key combination" method only works in computers but not in mobile phones. Some other participants who use special keyboards to input Unicode passwords said this method is not so universal: "with different keyboards, the results of 'key combination' are not identical. Even within the same keyboard, the number of Unicode character that can be generated is limited".

Fig. 4.6: The keyboard's Unicode layout via the key combination method

4.3.5 The follow-up user study result: Participants' evaluation and opinion towards the tool

Participants were asked to rate various options in the survey. The lowest score is 1 and the highest score is 5. Detailed evaluation criteria and rating result were in the following table.

Table 4.8: Survey scores

Evaluation Criteria	**Average score**
Willingness to refer it to others	3.6
Convenience to use	4.5
Seamless of user interface	4.3
Security features	4.6
Installation process easiness	4.7

In general, in many participants' opinions, the tool had a seamless integration with the password input field. And it saved them a lot of time to input Unicode passwords with extensive security features. The installation process is also easy: just loading the extension.

The only criteria which have a relative low score is "Willingness to refer it to others". Some

participants who gave a low score explained that some of their friends were non-tech heavy users and they might not be so interested in using the tool.

In addition to requested to simply rate various aspects of the tool, participants were also required to write a personal review about the tool. The following contents are several key points extracted from all participants' review.

Tendency to continue using the tool. Several participants reported that they found it very meaningful in their involvement in this user study. Through their participation, they discovered a highly effective tool previously unknown to them. They expressed their intention to continue utilizing this Unicode password input helper tool in their daily activities beyond the conclusion of the study.

Unicode password creation based on native languages. In all participants' reviews, most of them mentioned that Unicode characters inside passwords they created were all characters based on their own first languages. They considered it more interesting and relaxing to create and use passwords based on their own language. Inside Unicode passwords that were shared from participants, we discovered that there was also a tight relation between the participant's nationality and Unicode characters' language type.

More confidence in the password strength and the account security. After create some Unicode passwords for their accounts, some participants felt a greater sense of security. One participant stated that his account would never be hacked by his naughty friends from then on because they had no knowledge that the password can be comprise of Unicode characters. Another participant commented that she could much better remember Unicode passwords in her memory so that she wouldn't need to write and save some of them in a notebook as a plaintext format, which might lead to a security risk.

4.3.6 The follow-up user study result: External technical implementation obstacles and recommendations to service providers

On the other hand, some participants pointed out that they had also encountered some unexpected inconvenience and trouble. Some participants mentioned that they failed to input and send Unicode passwords because they did not meet the password regular expression (Regex) requirements from some websites. Some participants found that they could not use the same Unicode password to login into one website and its corresponding mobile app.

Although the tool itself does not exhibit any critical issue from the current point of view, according to feedbacks from participants, it may not perform well on them due to some websites or applications' specific implementations.

Websites' Unicode password acceptance rate

According to recent recommendations regarding passwords, it is advocated for the inclusion of diverse character types and the acceptance of various symbols, numbers, and special characters in addition to letters [24][25]. Nevertheless, researchers found that only around 70% of sites do accept Unicode passwords [26]. Several legacy sites even only accept digit-only passwords [27]. Websites' acceptance for Unicode passwords is the prerequisite for using this tool. Those sites ought to update and modernize their authentication systems to meet current demands.

Character encoding difference

Until April, 2024, UTF-8 is used by 98.2% of surveyed web sites [28]. However, among websites whose language is Japanese, 3.9% of them use Shift JIS and 2.9% of them use EUC-JP [29]; among

websites whose language is Chinese, 3.3% of them use language specific character set [30], as shown in Table 4.9. Those websites whose character encoding set are not UTF-8 may encounter the following problems when using Unicode passwords.

Table 4.9: Relationship between website language and its encoding method

Website language	Non-UTF-8 usage ratio(%)
Japanese	6.8%
Chinese	3.3%
Korean	5.1%

Website and mobile application character encoding difference. The website and mobile application provided by the same service may exhibit disparities in character encoding sets, owing to the evolution of technology over time. For instance, while the old school website might adhere to a custom character encoding set for the backward capability, the contemporary mobile app is more likely to adopt the default and universally accepted UTF-8 encoding standard. Consequently, this incongruity in encoding protocols could potentially result in login failures for users who inadvertently create their passwords using Unicode characters on one platform, unaware of the encoding limitations present on the other platform.

Front-end and back-end character encoding difference. The front-end and back-end of the same service might utilize distinct character encoding sets. While many programming languages default to UTF-8 for character encoding, issues can arise when the back-end encounters form data from the front-end encoded in a different method [31]. This disparity could result in difficulties creating and inputting Unicode passwords, potentially leading to login failures.

To address the issue of character encoding differences between web and mobile platforms causing login failures for users who use Unicode characters in their passwords, consistent encoding is needed. Both the website and mobile application, the front-end and the back-end should use the same character encoding scheme, preferably UTF-8, which is widely supported and capable of representing most characters in the Unicode standard. This consistency eliminates the disparity in encoding protocols and prevents login failures due to encoding mismatches.

Regular expression checking

Some websites use regular expression (regex) to implement password strength requirements. For example, the following regex:

```
/^[\w\.(!@#$\%&)]{6,20}$/
```

defines a range of characters that are allowed and required in a string, with a length constraint from 6 to 20 characters. It includes alphabets (\w) and certain special characters (!@#$%&). However, the regex does not include and allow Unicode characters.

Implementing password strength requirements with regular expressions is a common practice, but it's important to take the character set coverage of the regex into consideration. Restricting passwords to a limited set of characters might inadvertently discourage users from creating strong and unique Unicode passwords. It is difficult for users to create Unicode passwords due to the regex mentioned above.

To improve the situation and allow users to input Unicode characters while still maintaining security, it is recommended to modify the regular expression pattern to include a broader range of

characters. Here's a revised regex pattern that allows Unicode characters:

```
/^[\w\p{L}\p{N}!@#$\%&]{6,20}$/.
```

This pattern allows for Unicode characters while still enforcing a length constraint of 6 to 20 characters.

4.3.7 The follow-up user study: summary

The result of this user study indicated that Unicode passwords can increase password strength while keep memorability unchanged. The Unicode password input helper tool have received positive reviews. It solves several traditional drawbacks when inputting Unicode password and it helps users to create and input Unicode passwords easier. Meanwhile, a certain amount of online service providers still uses legacy technology which may be incompatible with this tool.

Chapter 5 Conclusion and future work

5.1 Conclusion

This research has demonstrated the significant advantages of incorporating Unicode characters into password creation to enhance security. By leveraging the tool developed in this research, users can easily create Unicode passwords that are not only more complex but also more resistant to brute-force attacks compared to traditional ASCII-based passwords due to extensive range of characters available in the Unicode standard. It has been proven that password strength can be increased without the sacrifice of password memorability using Unicode password. The experiments and user studies conducted as part of this research have provided valuable insights into the practical implementation and user acceptance of Unicode passwords. The proposed solution, which includes a browser extension to assist users in creating and managing Unicode passwords, has shown promise in improving both security and usability. User feedback has been generally positive, highlighting the tool's effectiveness in simplifying the process of inputting Unicode characters and enhancing password strength.

5.2 Future work

Spread the tool to more people. To maximize the benefit of our tool, its reach must be broadened. The tool can be upload to official web extension store to reach more users. It is also probable to utilize social media platforms and online communities to generate buzz and foster discussions. Meanwhile, more user feedbacks and usage statistics should be encouraged to shared to have a better insight of the tool's effect.

Acknowledgement

I would like to express my sincere gratitude to my supervisor, Tatsuya Mori, for his guidance, support, and invaluable feedback throughout the entire process of researching and writing this thesis. His expertise and encouragement have been instrumental in shaping the direction of my work. I would also like to acknowledge the countless researchers and scholars whose work has paved the way for this study. Their contributions to the field have been a constant source of inspiration and motivation.

Bibliography

[1] FIDO (Fast IDentity Online) Alliance. Fido alliance study reveals global password usage is down - yet its continued dominance is proving costly. https://fidoalliance.org/barometer-2022, 2022.

[2] Have i been pwned. https://haveibeenpwned.com.

[3] Have i been pwned. https://x.com/haveibeenpwned.

[4] https://en.wikipedia.org/wiki/List_of_Unicode_characters.

[5] Emoji password. The 8th annual shorty awards, https://shortyawards.com/8th/emoji-passcode.

[6] Lydia Kraus, Robert Schmidt, Marcel Walch, Florian Schaub, Christopher Krügelstein, and Sebastian Möller. Implications of the use of emojis in mobile authentication. In *USENIX SOUPS*, 2016.

[7] Tobias Seitz, Florian Mathis, and Heinrich Hussmann. The bird is the word: A usability evaluation of emojis inside text passwords. In *Human – Nature, OzCHI 2017, Nov 28 - Dec 1, Brisbane, Australia*, 2017.

[8] FIDO Alliance. Open authentication standards more secure than passwords. https://fidoalliance.org.

[9] How fido addresses a full range of use cases. https://media.fidoalliance.org/wp-content/uploads/2022/03/How-FIDO-Addresses-a-Full-Range-of-Use-Cases.pdf, 2022.

[10] Lucas Ropek. Google rolls out passkeys to (eventually) kill passwords. gizmodo.com, as of Oct. 10, 2023, 2023.

[11] Kate O'Flaherty. Apple to kill passwords with gamechanging new face id move. forbes.com, as of Oct. 10, 2023, 2021.

[12] Nina Bindel, Cas Cremers, and Mang Zhao. Fido2, ctap 2.1, and webauthn 2: Provable security and post-quantum instantiation. In *Proc. SP*, 2023.

[13] Leona Lassak, Elleen Pan, Blase Ur, and Maximilian Golla. Why aren't we using passkeys? obstacles companies face deploying fido2 passwordless authentication. In *USENIX 33rd*, 2023.

[14] Michelle Castillo. New apple iphone app proves just how hard it is to kill the online password. *CNBC*, 2024.

[15] Sunpreet S. Arora, Saikrishna Badrinarayanan, Srinivasan Raghuraman, Maliheh Shirvanian, Kim Wagner, and Gaven Watson. Avoiding lock outs: Proactive FIDO account recovery using managerless group signatures. Cryptology ePrint Archive, Paper 2022/1555, 2022. https://eprint.iacr.org/2022/1555.

[16] Bitwarden. https://bitwarden.com.

[17] https://www.ipqualityscore.com/ip-reputation-check.

[18] SecLists. https://github.com/danielmiessler/SecLists/tree/master/Passwords/Common-Credentials.

[19] Hashcat. https://github.com/hashcat/hashcat.

[20] thc-hydra. https://github.com/vanhauser-thc/thc-hydra.

[21] https://github.com/praetorian-inc/Hob0Rules/blob/master/wordlists/rockyou.txt.gz.
[22] https://github.com/openwall/john/blob/bleeding-jumbo/run/password.lst.
[23] https://github.com/danielmiessler/SecLists/blob/master/Passwords/Common-Credentials/10-million-password-list-top-100000.txt.
[24] P Grassi, Michael E Garcia, and James L Fenton. Digital identity guidelines. Technical report, NIST Special Publication 800, 2017.
[25] The Open Web Application Security Project. https://cheatsheetseries.owasp.org, 2024.
[26] Suood Alroomi and Frank Li. Measuring website password creation policies at scale. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23)*, pp. 3108–3122. Association for Computing Machinery, New York, NY, USA, 2023.
[27] Ding Wang and Ping Wang. The emperor's new password creation policies. In *European Symposium on Research in Computer Security (ESORICS)*, 2015.
[28] https://w3techs.com/technologies/cross/character_encoding/ranking.
[29] https://w3techs.com/technologies/segmentation/cl-ja-/character_encoding.
[30] https://w3techs.com/technologies/segmentation/cl-zh-/character_encoding.
[31] Bonneau J. and Xu R. Of contrasenas, ~ !, and 密 character encoding issues for web passwords. In *Web 2.0 Security & Privacy*, 2012.