

Doodles: A Hash Visualization Scheme



By

Momna Saeed

Fall 2017 - MS(IT) - 00000204770

Supervisor

Dr. Syed Taha Ali

Department of Computing

A thesis submitted in partial fulfillment of the requirements for the degree
of Master of Science in Information Technology (MS IT)

In

School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan.

(June 2021)

Approval

It is certified that the contents and form of the thesis entitled "Doodles: a Hash Visualization Scheme" submitted by MOMNA SAEED have been found satisfactory for the requirement of the degree

Advisor : Dr. Syed Taha Ali

Signature:  _____

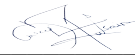
Date: 20-Apr-2021

Committee Member 1: Mr. Muhammad Imran
Abeel

Signature:  _____

Date: 20-Apr-2021

Committee Member 2: Dr. Wajahat Hussain

Signature:  _____

Date: 19-Apr-2021

Committee Member 3: Dr. Muhammad Latif Anjum

Signature:  _____

Date: 21-Apr-2021

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis entitled "Doodles: a Hash Visualization Scheme" written by MOMNA SAEED, (Registration No 00000204770), of SEECs has been vetted by the undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/M Phil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature: _____  _____

Name of Advisor: Dr. Syed Taha Ali _____

Date: _____ **20-Apr-2021** _____

Signature (HOD): _____

Date: _____

Signature (Dean/Principal): _____

Date: _____

Dedication

Dedicated to my family and friends for their never-ending support, motivation, and love.

Certificate of Originality

I hereby declare that this submission titled "Doodles: a Hash Visualization Scheme" is my own work. To the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics, which has been acknowledged. I also verified the originality of contents through plagiarism software.

Student Name: MOMNA SAEED

Student Signature: *Momna*

Acknowledgment

First and Foremost, praises and thanks to Allah almighty for His blessings throughout my research work, to complete the research successfully, and without His guidance I could not have completed this task.

I would like to express my deep and sincere gratitude to Dr. Syed Taha Ali for guiding and supervising me to complete my thesis. His vision, sincerity and motivation have deeply inspired me. It is a great privilege and honor to work and study under his guidance. His efficient contributions helped me shape the thesis into its final form and I express my gratefulness for his sincere supervision all the way.

I am thankful to Dr. Wajahat Hussain, Dr. Muhammad Imran Abeel and Dr. Latif Anjum for being on my thesis guidance and evaluation committee. I am also thankful to Dr. Yousra Javed for being in my research team and mentoring me during this project. I am grateful to my research colleague Talha Nadeem for his kind support and valuable contribution to this project.

I offer my tribute to my Department at National University of Science and Technology NUST, for providing me opportunity to complete my study here.

I would like to thank my friends and fellows for their encouragement and motivation. I also acknowledge every participation done in process of completing this research work.

Last but not the least, I would like to acknowledge my parents who have always taken extra steps to ensure every comfort of life for me. I am thankful to my family for all the support and prayers. It is because of their dedication, warmth and encouragement which enabled me to reach where I am today.

Table of Contents

Approval	i
Acceptance Certificate	ii
Dedication	iii
Certificate of Originality	iv
Acknowledgment	v
Table of Contents	vi
List of Tables	viii
List of Figures	ix
Abstract	xi
Chapter 1 Introduction	1
1.1 Challenge	3
1.2 Problem Statement	3
1.3 Solution Statement	4
1.4 Research Impact	4
1.5 Model of Study	5
1.6 Thesis Organization	6
Chapter 2 Background and Literature Review	7
2.1 Area of Research	7
2.2 Role of Hashing	8
2.3 Hash Visualization	9
2.3.1 Existing Hash Visualization Schemes	9
2.4 Use of Doodles in Cryptography	11
2.5 Research Aim	13
Chapter 3 Research Methodology	14
3.1 Parts of Research	14
3.2 Existing Techniques	14
3.2.1 Base32	14
3.2.2 Base58	15
3.2.3 TFlag	16
3.2.4 Random Art	17

Table of Contents

3.2.5	Alpha Hash	19
Chapter 4	Proposed Scheme: System Design and Implementation	21
4.1	Doodles Hash System	21
4.2	Quickdraw dataset	21
4.3	System Implementation	22
Chapter 5	Usability Study & Design	27
5.1	Design Goals	27
5.2	Designing Hash Pairs	28
5.3	Online Study	29
5.4	Participant Demographics	37
Chapter 6	Results and Analysis	38
6.1	Data Preprocessing:	38
6.2	Results and Analysis	39
6.2.1	Factor wise impact	40
6.2.1.1	Gender:	40
6.2.1.2	Age Group:	41
6.2.1.3	Background:	42
6.2.1.4	Familiarity:	44
6.2.1.5	Experience:	45
6.2.2	Performance of Each Hash Comparison Scheme	46
6.2.2.1	Alpha Hash	46
6.2.2.2	Base-32	48
6.2.2.3	Base-58	49
6.2.2.4	Random Art	51
6.2.2.5	T-Flag	53
6.2.2.6	Doodle Hash	54
Chapter 7	Conclusion and Future Work	56
7.1	Additional Properties of Hash Comparison Schemes	56
7.2	Discussion	57
7.3	Conclusion	59
7.4	Future Work	61
	Bibliography	62

List of Tables

Table 1: Alpha Hash Summary.....	47
Table 2: Base32 Summary	48
Table 3: Base58 Summary	50
Table 4: Random Art Summary.....	51
Table 5: TFlag Summary.....	53
Table 6: Doodle Hash Summary.....	54
Table 7: Additional properties of schemes.....	56
Table 8: Summary of Hash schemes.....	60

List of Figures

Figure 1: Hash Function.....	2
Figure 2: Model of Study	5
Figure 3: Thesis Organization	6
Figure 4: Areas of Research	7
Figure 5:Random Art	10
Figure 6: TFlag	11
Figure 7:Doodle example.....	12
Figure 8: Base32 Sample Conversion.....	15
Figure 9: TFlag Sample Conversion	16
Figure 10: TFlag Implementation using Atom	17
Figure 11: Random Art Sample 1	17
Figure 12: Random Art Sample 2	18
Figure 13: Random Art Sample 3	18
Figure 14: Random Art Sample 4	19
Figure 15: Alpha Hash Sample 1	19
Figure 16: Alpha Hash Sample 2.....	19
Figure 17: Quickdraw Dataset Input[32].....	21
Figure 18: Example doodles of quick draw dataset	22
Figure 19: Snapshot of image conversions using python.....	23
Figure 20: Images for doodle hash generation	23
Figure 21: Input alphanumeric hash string.....	24
Figure 22: Snapshot of doodle hash conversion	24
Figure 23: Doodle Hash output string.....	25
Figure 24: Generated Doodle Hash Sample 1	25
Figure 25: Generated Doodle Hash Sample 2	25
Figure 26: Generated Doodle Hash Sample 3	26
Figure 27: Generated Doodle Hash Sample 4	26
Figure 28: Perceptual Image Diff-Snapshot case1.....	29
Figure 29: Perceptual Image Diff-Snapshot case2.....	29
Figure 30: Perceptual Image Diff-Snapshot case3.....	29
Figure 31: Usability Study Snapshot 1.....	31
Figure 32: Usability Study Snapshot 2.....	31
Figure 33: Usability Study Snapshot 3.....	32
Figure 34: Usability Study Snapshot 4.....	32
Figure 35: Usability Study Snapshot 5.....	33
Figure 36: Hash Comparison Snapshot 1-TFlag.....	33
Figure 37: Hash Comparison Snapshot 2-Base32	34
Figure 38: Hash Comparison Snapshot 3-Alpha Hash.....	34
Figure 39: Hash Comparison Snapshot 4-Base58	35

Figure 40: Hash Comparison Snapshot 5-Doodle Hash.....	35
Figure 41: Hash Comparison Snapshot 6-Random Art.....	36
Figure 42: Usability Study Snapshot 6.....	36
Figure 43: Usability Study Snapshot 7.....	37
Figure 44: Usability Study Snapshot 8.....	37
Figure 45: Base-32 (Easy) with outliers.....	38
Figure 46: Base-32 (Easy) without outliers.....	39
Figure 47: Count of Participants by Gender.....	40
Figure 48: Response Time (secs) by Gender.....	41
Figure 49: Average of Response Time (secs) by Age Group.....	42
Figure 50: Count of Participants by Age Group.....	42
Figure 51: Count of Participants by Background.....	43
Figure 52: Average of Response time by Background.....	44
Figure 53: Count of Participants by Familiarity.....	44
Figure 54: Average of Response Time by Familiarity.....	45
Figure 55: Count of Participants by Experience.....	45
Figure 56: Average of Response Time by Experience.....	46
Figure 57: Alpha Hash Response Time.....	47
Figure 58: Alpha Hash Accuracy.....	48
Figure 59: Base-32 Average Time (secs).....	49
Figure 60: Base-32 Accuracy.....	49
Figure 61: Base-58 Average Time(secs).....	50
Figure 62: Base-58 Accuracy.....	51
Figure 63: Random Art Average Time(secs).....	52
Figure 64: Random Art Accuracy.....	52
Figure 65: T-Flag Average Time.....	53
Figure 66: T-Flag Accuracy.....	54
Figure 67: Doodle Hash Average Time(secs).....	55
Figure 68: Doodle Hash Accuracy.....	55
Figure 69: Scheme wise Average Response Time.....	57
Figure 70: Scheme wise Average Accuracy.....	58
Figure 71: Scheme Accuracy by Average Response Time.....	59

Abstract

Several authentication techniques and security protocols require users to compare hash strings in different forms like cryptographic keys, addresses, and identifiers. As these hash values are long sequences of digits and alphanumeric strings, there is quite a chance that humans may have problems in precise comparisons of the hash values. Also, considerable time and effort are required for these strings.

If the hash values are not compared properly, this raises a high probability of a man-in-the-middle attack. The adversary performing such attacks can take advantage of human limitations for instance users are slow and inaccurate while comparing long pointless strings. These constraints increase the negative effects on the security of verification and validation of certain applications and user authentication. To perform secure communication, there is a need for a secure and usable mechanism for hash representations. In this research study, the textual and alphanumeric sequences of different hash forms like cryptographic keys and addresses are converted to visual fingerprints that make it easy for humans to perform comparisons. Graphical representations are a promising substitute for hash comparisons because humans can speedily identify dissimilarities in graphical images.

We propose Doodle images as a hash visualization technique. The hash value is represented as a sequence of doodles that is easy to visualize as well as compare. It is easy for any technical or non-technical user to compare and authenticate the doodle hash images as compared to traditional hashes. This research work focuses on the implementation of various visualization techniques and performing a comparative analysis. The goal is to determine which conversion method provides the accurate results of comparison as well as which one is fastest and caters the human limitations. So, these techniques are compared and evaluated with each other and with the proposed doodle hash representation. This is done by performing an online usability study with different participants.

Keywords: *Hash Visualization, Doodles, Hash Comparison, Visual Hash schemes*

Chapter 1 Introduction

Cybersecurity has become an integral part of almost all domains on the internet. Whether it is communications or data storage on all types of networks. It encompasses numerous hardware and software technologies that work together to ensure the security and integrity of the network. Of all the cybersecurity methods available for use, encryption is amongst the most widely used.

Encryption is the method by which an algorithm is used to convert readable information and make it incomprehensible for unauthorized individuals. This is a branch of cryptography and is used to protect very sensitive data such as passwords and pin codes with the help of encoding. Once the data is converted into its encoded form, it can only be deciphered with a unique key. The widely used types of encryptions are symmetric and asymmetric encryptions. The only difference between the two is whether or not the same key is needed for decryption. In symmetric encryption, the similar type of key is used for the decryption and encryption of the sent and received information. For this purpose, it is therefore essential that a secure method is used to share the key between the sender and recipient. In asymmetric encryption, a key-pair is used. This means that a different key is used for sending data and a separate key is used for receiving that data. The keys are called public and private keys.

Maintaining the integrity of the data being transmitted ensures that the data has not been changed in any way. This means that the data has not been modified, tampered with, or has been corrupted. However, unauthorized users may gain access to the data by malware or through human errors. In this case, the concept of hashing is brought into consideration.

The process of converting a given key to another value is known as Hashing. A special type of function, called a hash function, is used to assign a unique value based on a mathematical algorithm. The output of a hash function is called a hash value[1]. By comparing hashes created at two different instances, it is possible to determine whether

the original data is the same or if it has been tampered with. Hashing can only let the users know if the data has been modified or not because if upon comparison, the hash values turn out to be different, it means that the information should not be trusted as valid.

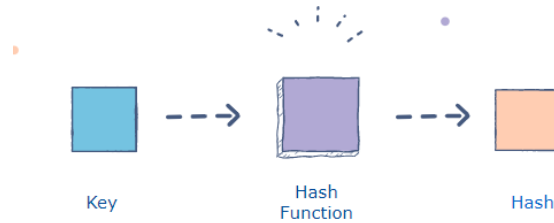


Figure 1: Hash Function

The process of hashing is used for the implementation of hash tables. Hash tables are used to save key and value pairs in the form of an array through which its elements can be accessed with the help of its indices. The hash values generated by the hash function act as the indices for the keys. There are various types of hash functions relying on the level of complexity required and the quantity of elements. Hashing is also associated with cryptography as it can be used in data encryption. In a nutshell, hashing is used to validate the integrity of the data under consideration.

The hash values generated by hash functions can be very long and complex depending on the type of application and complexity of the hash function. The only method of verification of the hash values is by comparison of the keys performed by humans. During the comparison process, human errors can occur as it is a very complex and sensitive procedure. Even the slightest error can render the data faulty. To ease the process of performing comparisons, some visualization schemes have been proposed to aid in faster and easier comparing processes.

The goal of visualization schemes is to act as a means of representing data in a more comprehensible form. In this case, the long and complex hash strings can be represented in the form of commonly known words or phrases, sentences, or even images. The purpose of this representation is to aid in the improvement of the usability

of hash values. Many visualization schemes exist that have shown positive results in this regard.

This research aims to first implement the use of doodles as a visualization scheme and then compare its usability to existing schemes. Doodles are random hand-drawn images. Instead of having to compare words or long strings, comparing doodles may be a faster and easier way of comparing hash values which can, in turn, speed up and simplify the authentication process.

1.1 Challenge

Hash comparisons by humans are required for the successful accomplishment of numerous security protocols. As these hash values are long sequences of digits and alphanumeric strings, there is quite a chance that humans may have problems in precise comparisons of the hash values. Also, considerable time and effort are required for these strings. To increase the usability of comparisons, the hash strings are converted to different representations like words, hexadecimal, sentences, and images. Doodle images can be used as hash visualization methods as these are easier for users to memorize and recognize thus decreases the cognitive load on the user.

1.2 Problem Statement

Several authentication techniques and security protocols require users to compare hash strings in different forms like cryptographic keys, addresses, and identifiers. If these strings are not compared properly, this raises a high probability of a man-in-the-middle kind of attack. The adversary performing such attacks can take advantage of human limitations for instance. Users are slow and inaccurate while comparing long pointless strings. These human limitations increase the negative effects on the security of verification and validation of certain applications and user authentication. To perform secure communication, there is a need for a secure and usable mechanism for hash representations.

1.3 Solution Statement

The textual and alphanumeric sequences of different hash forms like cryptographic keys, addresses, and fingerprints are converted to visual fingerprints that make it easy for humans to perform the comparison. This hash visualization technique improves real-world security and aids in the validation and verification of certain applications. Also reduces the vulnerability of man-in-the-middle attacks. Graphical representations are promising alternatives for fingerprint comparison because humans can quickly identify dissimilarities in images.

1.4 Research Impact

With the help of visual hash forms, it will make it easier for humans to perform comparisons quickly and accurately. The overall effect of this will be that security credential verification times will be dramatically shortened and added with lesser chances of human errors. This research will also determine which form of hash strings is preferred by most practitioners.

1.5 Model of Study

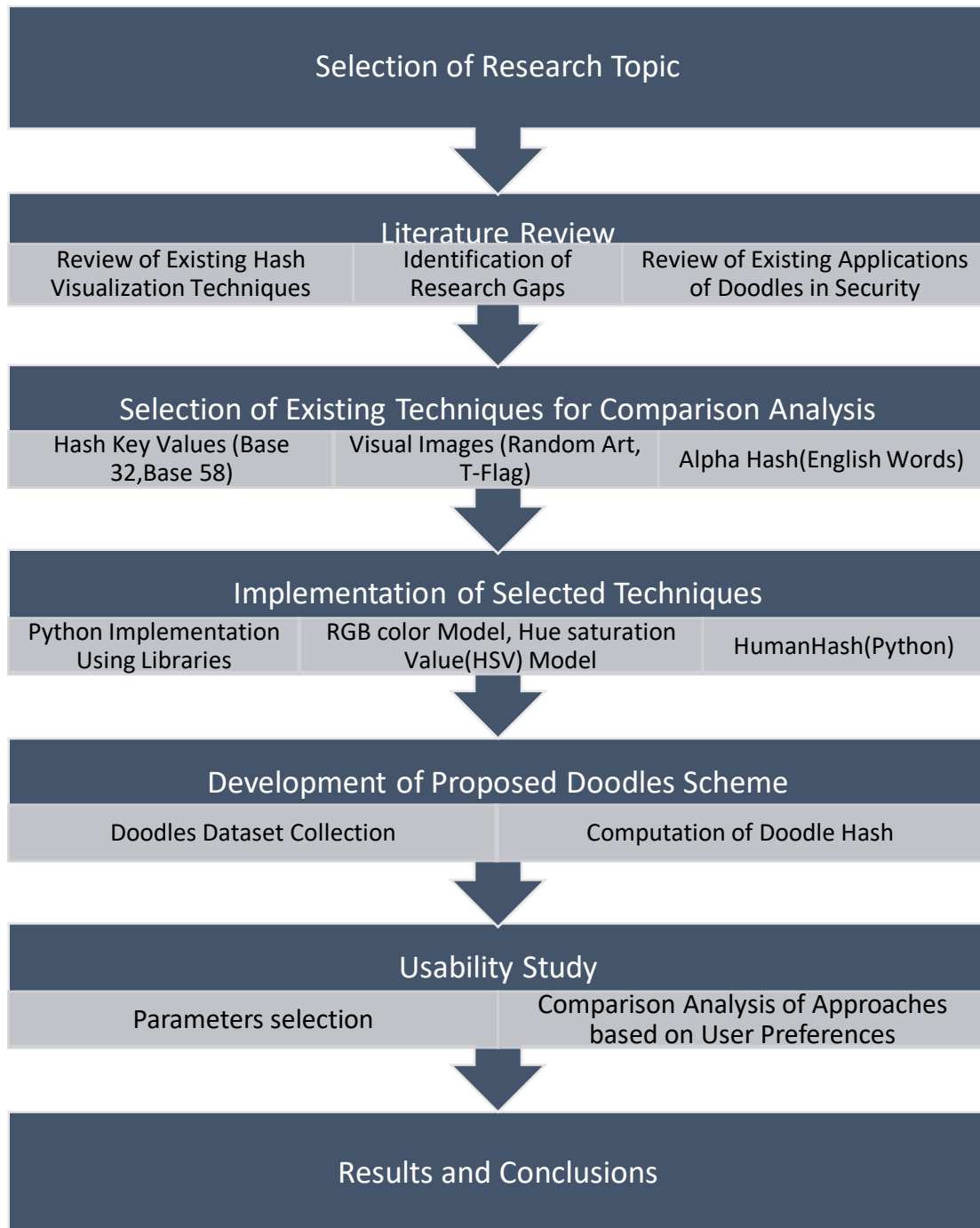


Figure 2: Model of Study

1.6 Thesis Organization

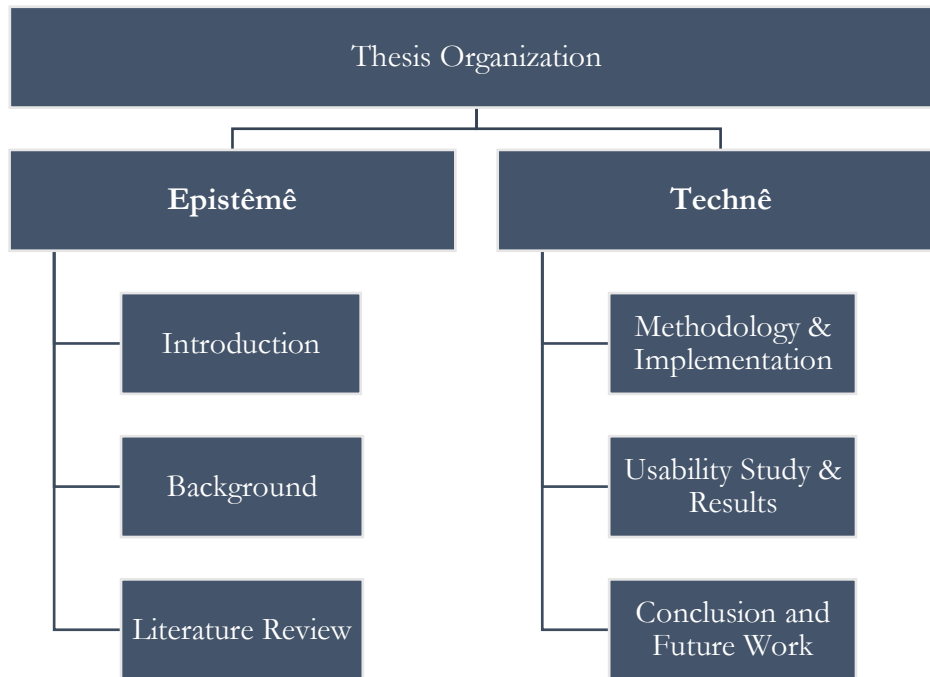


Figure 3: Thesis Organization

Chapter 2 Background and Literature Review

2.1 Area of Research

A detailed literature review has been conducted to identify studies related to existing hash visualization schemes. Studies show that there have been attempts made to improve the methods involving hash key comparisons. All the techniques studied in the available literature circulate the domain of cryptography and data integrity. The goal of this research is to identify the gaps in existing techniques and pave the way for a smoother and more efficient method of hash key comparisons. The provided literature is divided into the following subsections as shown in the figure below.

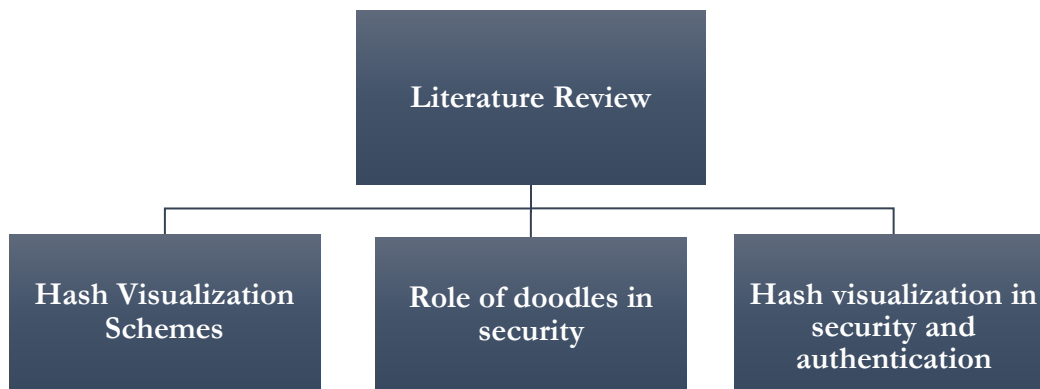


Figure 4: Areas of Research

2.2 Role of Hashing

Hashing is associated with data integrity[2]. In almost all forms of digital communication, data integrity ensures that the data has not been altered or tampered with by any means. In ideal scenarios, only authorized users are allowed access to the data. Occasionally, there are times when unforeseen incidents can result in a data breach or the tampering of data via unauthorized users, malware, and mostly because of human errors.

Hashing techniques are most commonly used to maintain data integrity. A hash is simply a unique identifier or key which is generated by applying a hash function to the data. The integrity of the data is verified by comparing the hash keys generated at the time of transmission and reception of the data. If the hash keys are the same, this means that the integrity of the data remained valid. If the two hash keys do not match each other, this means that the data has been tampered with.

There are a variety of hashing algorithms that generate different types of hash keys for comparisons. The key difference between these different algorithms is the type of hash keys that they generate.

MD5 is a popular hashing algorithm that generates hash keys in the 128-bit hash format. Rather than binary representations, the hash keys generated using this algorithm are displayed in hexadecimal formats. Using this format, the hash keys are displayed as 32 hexadecimal characters rather than 128 bits.

SHA, or Secure Hash Algorithm, is another type of hashing algorithm which has multiple variations. Taking the example of SHA-1, this is quite similar to the MD5 hashing technique, but it creates 160-bit hashes instead of 128-bit hashes.

As an example, let us take the phrase, 'The quick brown fox'. Applying both the mentioned hashing algorithms on this phrase, we will get the following results:

MD5: '2e87284d245c2aae1c74fa4c50a74c77'

SHA-1: 'ced71fa7235231bed383facfdc41c4ddcc22ecf1'

In both cases, the phrase of words has been converted into hexadecimal values with the key difference being the length of the generated hash keys[3]. In an ideal scenario, these are the keys that are to be compared by human intervention to ensure the integrity of the data.

2.3 Hash Visualization

As we know, visualization is the process of transforming data into a form for much easier interpretation. In the case of hashing, hash visualization is the process of representation of hash keys in graphical forms[4].

The purpose of this is to minimize the chances of human errors that may occur during the comparison process of hash keys[5]. People need to compare meaningless key fingerprints[6]. These key values can be up to 32 hexadecimal bits depending on the type of hashing algorithm used. It has been identified that humans are slow and unreliable at memorizing such long values [7]. This can also be understood based on the fact that most of us have difficulty in memorizing long passwords. On the other hand, if the passwords comprise commonly used words and phrases, then there is the risk of vulnerability[4].

Keeping these issues in mind, if these hash values are converted into visual strings and images[8], it is easier for humans to identify and compare hash values more reliably and efficiently [9].

2.3.1 Existing Hash Visualization Schemes

Multiple efforts have been made to develop such a hash visualization scheme that outperforms its predecessors in efficiency and utilization. All these schemes have the same goal of assisting humans in the quick and easy comparison of hash keys to reduce the possibility of human errors. In this section, existing hash visualization schemes will be introduced.

Hexadecimal encoding[8] is being used for a long time for comparisons because it enables users to reduce long and error-prone hash keys into a short sequence of letters and digits. Examples of algorithms that use this method of conversion are Base32 and Base58[10].

Base32 uses the reduced set of digits namely 2-7 and capital letters A-Z. The benefits of using Base32 are that it reduces the confusion among similar-looking digits (0 and 8)[11]. Similar to base32, base58 utilizes 58 characters instead of 32 characters[8]. This improves the reliability of data integrity in terms of cryptography[6].

Base58[12][10] is a widely used encoding format that is also used for bitcoin[13][14] and other cryptocurrencies because it offers effective compression, easy readability, and error diagnosis. In the case of bitcoin[10], a Base58Check[12] is appended in the base58 encoding to effectively check for errors in transmission[15].

The next form of easier visualization of hash keys is in the form of common words. This means that the hard-to-process hash keys are converted into words[8] that can be easily processed. This form of visualization is called an Alpha Hash or simply, English Words [11].

Another visualization scheme converts the hash keys into images of randomly generated art. This algorithm is known as Random Art [7]. These images are generated with the help of complex arithmetic expressions that assign values to individual pixels in the image.



Figure 5:Random Art

Another approach that involves more sequenced images is T-Flag[16]. Unlike its predecessor Flag, T-Flag converts the hash keys into an image consisting of 8 randomly placed rectangular blocks of colors. These colors have been specifically selected to ensure that even individuals who are affected by colorblindness can easily differentiate between the colors during the comparison process.



Figure 6: TFlag

2.4 Use of Doodles in Cryptography

All computer users are required to remember a fair amount of login and password details. It becomes cumbersome to the extent that the use of password management systems has been given birth. Passwords can be stored or written down but that gives rise to security risks and vulnerability. To tackle this ever-growing problem, the use of hand-drawn images, also called doodles, has been proposed in replacement to conventional passwords[17].



Figure 7:Doodle example

The concept is that instead of having to memorize passwords for signing into platforms, users can set a hand-drawn doodle[18][19] as their password and simply draw it to log in. Such passwords are also referred to as ‘passdoodles’[20]. According to the findings of this research, users perceive passdoodles as easier to remember than conventional alphanumeric passwords[19]. The use of doodles allows for a better recall rate as compared to alphanumeric passwords[19][21]. The ease of use is also higher compared to conventional passwords[4].

As pointed out in another research, doodles are an extension of hand-written passwords[22].The method of registering the doodles is quite simple. An initial doodle training session would encourage the user to draw their doodle[19] multiple times so that the system can register the user inputs and train the model based on select features such as the unique shape and movement of the users drawing[21]. This helps the system in quickly distinguishing the user’s doodle from other doodles[18] registered in the system against other users.

The use of Key Fingerprint Generators, or KFG’s, has somewhat improved the process of key comparison by converting the long strings into images or shorter texts for comparison. A group of researchers built on the idea of KFG’s and developed CEAL, or CrEdential Assurance Labeling[23]. The key difference between CEAL and traditional KFG’s is that it eliminates the need for hand-generated images and also generates unique images that can be easily distinguished by individuals. Their research has proved that a CEAL-trained Visual KFG is considerably superior to state-of-the-art solutions, in terms of human accuracy, entropy and speed of evaluation.

A more prominent study carried out by researchers involved a visual hash technique encoding 60 bits by means of Colors, Patterns, and Shapes, or CLPS for short[24]. In this study, Base32 hash keys were converted into sets of images. These images were then presented to various individuals who were then asked to compare the two images and verify if they were similar or not. Afterward, they were then asked to compare the Base32 hash keys. In both cases, the times taken to compare the samples were noted for each of the individuals. The results of their study concluded that the users took significantly lesser time to compare CLPS pictures than they did to compare Base32 characters for both easy pairs and hard pairs. This demonstrated that the users were more competent at comparing CLPS images in contrast to comparing Base32 hash keys.

Throughout this research, it has been observed by the researchers that the test subjects found the use of doodles much easier as compared to conventional passwords. It has also proved to have greater anonymity[17]. Participants were also able to remember their doodles just as precisely as conventional passwords[20].

2.5 Research Aim

The use of doodles for authentication purposes can be found in the existing literature[4][17][19]. However, the use of doodles as hash key visualizations has not been explored. Our research aims at applying the ease-of-use and convenience of doodles for quick and efficient comparison of hash keys. This method is expected to reduce human errors that can occur during the comparison of hash keys while maintaining the integrity of the underlying data.

Chapter 3 Research Methodology

3.1 Parts of Research

There are two parts to this research. The first part consists of generating hash keys using various hash visualization methods. The second part of this research consists of designing and conducting the survey which will, in turn, give us detailed feedback on how long different users take to compare the different encrypted hash keys. Using the results of the survey, we will compare the time taken in comparing doodles versus the time taken in comparing hash keys in multiple forms.

3.2 Existing Techniques

The mentioned survey will contain hash keys that will be generated using different encryption technologies. The users will then be asked to compare the hash keys whereas the time taken by each user for each selection will be recorded. To make this research as extensive as possible, the following encryption technologies will be used for this research:

1. Base32
2. Base58
3. T-Flag
4. Random Art
5. Alpha Hash

Further, the technologies used for the implementation of these encryption methods are discussed below. They are also explained with images to help the reader understand.

3.2.1 Base32

Base32 encryption has been implemented in our research using the readily available open-source programming language, python. After setting up the python environment on our computer, the Base32 library version 1.0.2 was downloaded and installed in the

Python environment. This library enables users to generate, encode, and decode base32 strings. We used this library to generate encrypted hash strings that we later incorporated into our user survey for comparisons.



The image shows a screenshot of a web-based tool for Base32 conversion. It is divided into two main sections: 'Input' and 'Output'.
In the 'Input' section, the text '1f42a19aeb048d4fe9856807033a00adc1239d2d' is entered. To the right of the input field, it displays 'length: 40' and 'lines: 1'.
In the 'Output' section, the converted string 'GFTDIMTBGE4WCZLCGA2DQZBUMZSTSOBVG4DANZQGMZWCMQMFSGGMJSGM4WIMTE' is shown. To the right of the output field, it displays 'time: 1ms', 'length: 64', and 'lines: 1'. There is a small icon of a pencil and an 'x' next to the 'Output' label.

Figure 8: Base32 Sample Conversion

In the image above, a sample conversion is shown for a string being encrypted using base32 encryption. The length of the output string is reminiscent of the length of the input string. In this particular example, the input string has 40 characters, and the output string has 64 characters.

3.2.2 Base58

Similar to the Base32 library, the python library package for base58 was installed beforehand. Once the python environment was configured, the library enabled us to generate hash keys in the base58[10] encryption. These generated hash keys were then added to the user survey. The base58 encryption[12] was created by Satoshi Nakamoto

which consists of 58 alphanumeric characters [25]. The goal in mind for this innovation was to be able to convert frequently used private keys into a format that was easy to use and share by everyone. Other than its use in bitcoins and cryptocurrency[13], base5[14]8 is also used to manage password keys and as well as short URLs. The purpose of Base58[14] is to represent large numbers in a short format while avoiding easily misinterpreted characters. Another interesting fact about base58 is that it does not use symbols that may create confusion during the process of comparisons such as 0's and O's.

3.2.3 TFlag

TFlag is an improved form of simple flag representation. This form of encryption falls under the category of visual representations. The key difference between the simple flag and TFlag[16] representation is that TFlag incorporates the use of 6 distinct colors that can easily be identified by individuals with visual impairments.

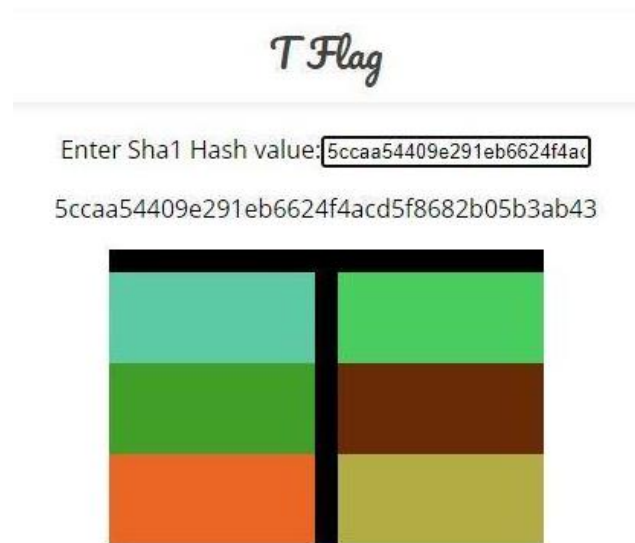


Figure 9: TFlag Sample Conversion

In the sample image above, the Sha1 Hash value is converted into the TFlag image. TFlag can be generated using SHA1 or MD5 [26]. In our case, we first encrypted the

hash key using SHA1. then we substring the hash to obtain a 6-digit hex code. Next, the hex codes are used to represent the various colors in the TFlag visual representation.

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

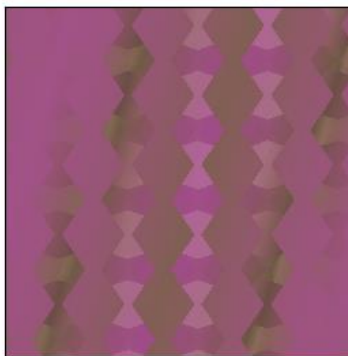
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\tflag 2\what-the-splash-starter2> npm start
<
  saga-splash@0.1.0 start D:\tflag 2\what-the-splash-starter2
> set PORT=8082 && react-scripts start
  
```

Figure 10: TFlag Implementation using Atom

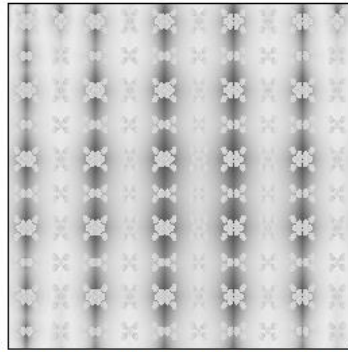
3.2.4 Random Art

Just as the name suggests, random art is a visual representation of hash keys with the help of randomly generated abstract art[7]. The process of art generation is that the hash key seeds a random number generator., the number generator constructs a mathematical formula. The formula in turn then calculates the color of each pixel in the image. As this process is algorithmic, the same key will generate the same image on each run. This algorithm was implemented in python for image generation[27].



4605b696d9d31a56034545048ea2b80d4466d933

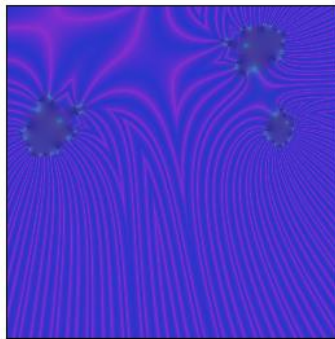
Figure 11: Random Art Sample 1



4605b696d9d13a56034545048ea2b80d4646d933

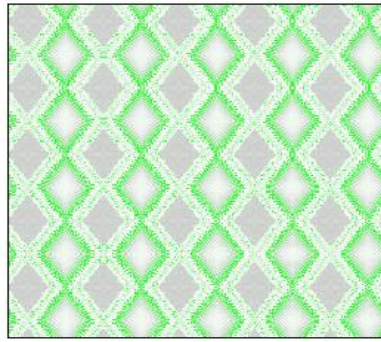
Figure 12: Random Art Sample 2

In the two figures above, the same hash string, with minor changes in the characters, is used on two different occasions to generate random art images[28]. The similarity in the string is reflected in the images having similar patterns with minor changes. In the two figures below, the random art images are generated from two completely different hash strings that produce two distinct random art images.



2fd4e1c67a2d28fced849ee1bb76e7391b93eb12

Figure 13: Random Art Sample 3



9d2278759ae24698b1345525bd53358b

Figure 14: Random Art Sample 4

3.2.5 Alpha Hash

This is one of the most basic encryption schemes in this research. An algorithm converts the hash keys into readable but randomized words. The method for generating an alpha hash is that the algorithm first compresses the hash key into a fixed length. Then, all of the resulting bytes of the compressed form are mapped to a predefined word from a library. This algorithm is consistent which means that the same hash key will result in the same alpha hash result.

```
In [1]: ▶ import humanhash
-----
In [2]: ▶ digest = '5ccaa54409e291eb6624f4acd5f8682b05b3ab43'
          ▶ humanhash.humanize(digest)
Out[2]: 'low-ten-butter-lake'
```

Figure 15: Alpha Hash Sample 1

```
In [1]: ▶ import humanhash
-----
In [2]: ▶ digest = 'ffe6d2df3d57607b96927a3407bee28b3c642adc'
          ▶ humanhash.humanize(digest)
Out[2]: 'cold-fix-beryllium-charlie'
```

Figure 16: Alpha Hash Sample 2

The above two figures show how a hash string is converted into Alpha Hash encryption. The 'humanhash' library is imported into the python environment and the passed hash string is converted into an alpha hash string which is a set of random human-readable words[29].

Chapter 4 Proposed Scheme: System Design and Implementation

4.1 Doodles Hash System

An application to generate doodle images as a conversion from alphanumeric hash strings to doodle hash.

4.2 Quickdraw dataset

The Quick Draw Dataset is a collection of 50 million drawings consisting of 345 categories[30], generated by the contribution of Quick Draw game players around the globe.

This game is built using machine learning. Users draw, and a neural network attempts to identify drawing. This dataset is open-sourced and being used for neural network training and other Artificial Intelligence and Data Analysis projects. These doodles drawings are very beneficial to various developers and researchers[31]. This dataset is being used for training the Sketch-RNN (recurrent neural network) model. The collection is accessible on Google Cloud Storage[32].

The quick draw game is created by Jonas Jongejan (web developer), Henry Rowley (Machine Learning Researcher), Takashi Kawashima, Nick Fox-Gieg and Jongmin Kim with the team at Google Creative Lab and Data Arts Team.



Figure 17: Quickdraw Dataset Input[32]

The quickdraw dataset is selected for our implementation because of these reasons:

1. It is the largest doodle dataset.
2. It is publicly available as open source.
3. This dataset is in use for current research projects in the AIML and Data Analysis domain.
4. As the doodles are hand-drawn by various users, they are easy to identify and more understandable for humans.



Figure 18: Example doodles of quick draw dataset

4.3 System Implementation

The doodle hash implementation is done using python modules and libraries in Python 2.7 version and Windows 10 operating system. The dataset that we used is comprised of more than 2500 doodle images.

The google draw dataset is transformed from category labels to hash values using python script.

```
Command Prompt
Microsoft Windows [Version 10.0.18362.1256]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\MOMNAH>cd..
C:\Users>cd..
C:\>cd C:\Python27
C:\Python27>python base64_rename_files.py
['.png', '0.png', '00.png', '01.png', '02.png', '03.png', '04.png', '05.png', '06.png', '07.png', '08.png', '09.png', '0A.png', '0B.png', '0C.png', '0D.png', '0E.png', '0F.png', '0G.png', '0H.png', '0I.png', '0J.png', '0K.png', '0L.png', '0M.png', '0N.png', '0O.png', '0P.png', '0Q.png', '0R.png', '0S.png', '0T.png', '0U.png', '0V.png', '0W.png', '0X.png', '0Y.png', '0Z.png', '1.png', '10.png', '11.png', '12.png', '13.png', '14.png', '15.png', '16.png', '17.png', '18.png', '19.png', '1A.png', '1B.png', '1C.png', '1D.png', '1E.png', '1F.png', '1G.png', '1H.png', '1I.png', '1J.png', '1K.png', '1L.png', '1M.png', '1N.png', '1O.png', '1P.png', '1Q.png', '1R.png', '1S.png', '1T.png', '1U.png', '1V.png', '1W.png', '1X.png', '1Y.png', '1Z.png', '2.png', '20.png', '21.png', '22.png', '23.png', '24.png', '25.png', '26.png', '27.png', '28.png', '29.png', '2A.png', '2B.png', '2C.png', '2D.png', '2E.png', '2F.png', '2G.png', '2H.png', '2I.png', '2J.png', '2K.png', '2L.png', '2M.png', '2N.png', '2O.png', '2P.png', '2Q.png', '2R.png', '2S.png', '2T.png', '2U.png', '2V.png', '2W.png', '2X.png', '2Y.png', '2Z.png', '3.png', '30.png', '31.png', '32.png', '33.png', '34.png', '35.png', '36.png', '37.png', '38.png', '39.png', '3A.png', '3B.png', '3C.png', '3D.png', '3E.png', '3F.png', '3G.png', '3H.png', '3I.png', '3J.png', '3K.png', '3L.png', '3M.png', '3N.png', '3O.png', '3P.png', '3Q.png', '3R.png', '3S.png', '3T.png', '3U.png', '3V.png', '3W.png', '3X.png', '3Y.png', '3Z.png', '4.png', '40.png', '41.png', '42.png', '43.png', '44.png', '45.png', '46.png', '47.png', '48.png', '49.png', '4A.png', '4B.png', '4C.png', '4D.png', '4E.png', '4F.png', '4G.png', '4H.png', '4I.png', '4J.png', '4K.png', '4L.png', '4M.png', '4N.png', '4O.png', '4P.png', '4Q.png', '4R.png', '4S.png', '4T.png', '4U.png', '4V.png', '4W.png', '4X.png', '4Y.png', '4Z.png', '5.png', '50.png', '51.png', '52.png', '53.png', '54.png', '55.png', '56.png', '57.png', '58.png', '59.png', '5A.png', '5B.png', '5C.png', '5D.png', '5E.png', '5F.png', '5G.png', '5H.png', '5I.png', '5J.png', '5K.png', '5L.png', '5M.png', '5N.png', '5O.png', '5P.png', '5Q.png', '5R.png', '5S.png', '5T.png', '5U.png', '5V.png', '5W.png', '5X.png', '5Y.png', '5Z.png', '6.png', '60.png', '61.png', '62.png', '63.png', '64.png', '65.png', '66.png', '67.png', '68.png', '69.png', '6A.png', '6B.png', '6C.png', '6D.png', '6E.png', '6F.png', '6G.png']
```

Figure 19: Snapshot of image conversions using python

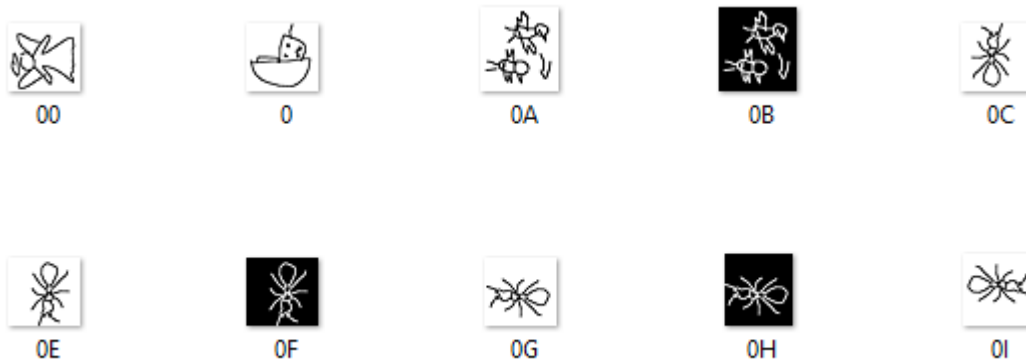


Figure 20: Images for doodle hash generation

After processing data, the doodle hashes are generated in the form of a pdf by using the script and fpdf library[33]. PyFPDF is a library for PDF document generation under Python. a

Proposed Scheme: System Design and Implementation

An input text file is provided having the alphanumeric hash string for which the doodle hash is to be generated.

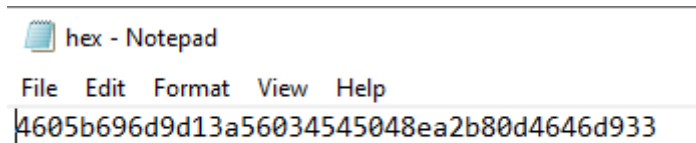
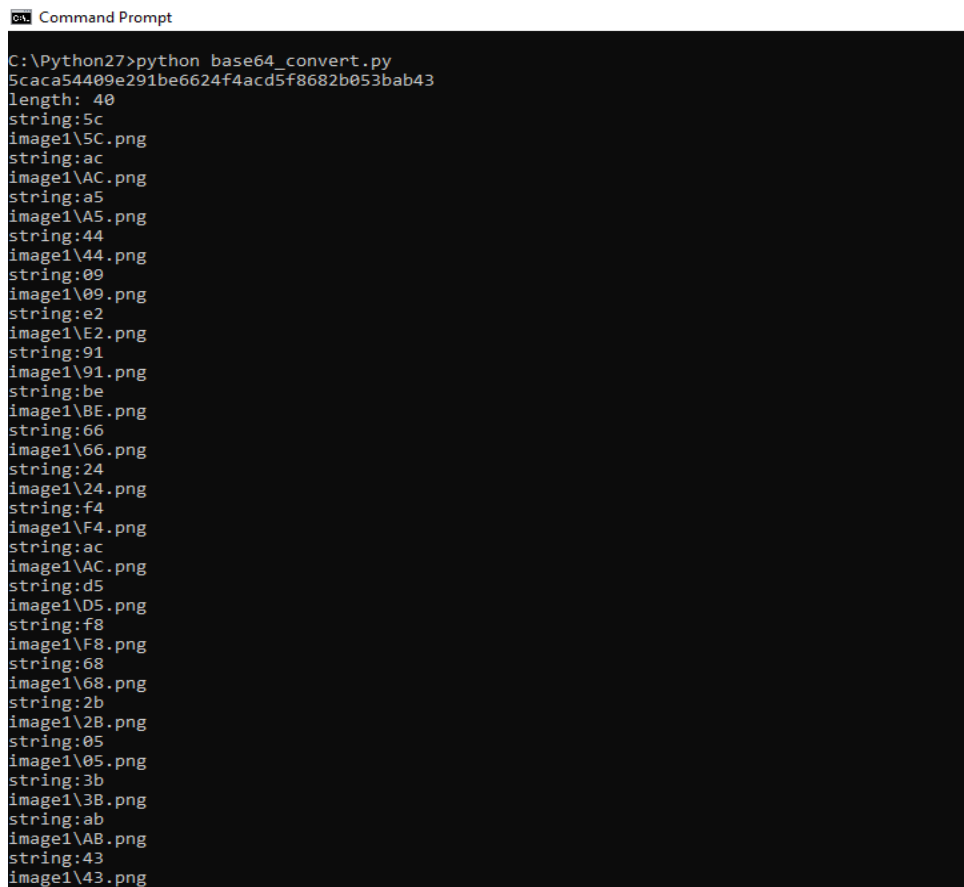


Figure 21: Input alphanumeric hash string

The python script reads from the text file and converts to a doodle hash output and a pdf is generated.



```
C:\Python27>python base64_convert.py
5caca54409e291be6624f4acd5f8682b053bab43
length: 40
string:5c
image1\5C.png
string:ac
image1\AC.png
string:a5
image1\A5.png
string:44
image1\44.png
string:09
image1\09.png
string:e2
image1\E2.png
string:91
image1\91.png
string:be
image1\BE.png
string:66
image1\66.png
string:24
image1\24.png
string:f4
image1\F4.png
string:ac
image1\AC.png
string:d5
image1\D5.png
string:f8
image1\F8.png
string:68
image1\68.png
string:2b
image1\2B.png
string:05
image1\05.png
string:3b
image1\3B.png
string:ab
image1\AB.png
string:43
image1\43.png
```

Figure 22: Snapshot of doodle hash conversion



Figure 23: Doodle Hash output string

Below are some examples of input hash strings and their converted doodle hashes generated by using this system.

Input String 1:

5caca54409e291be6624f4acd5f8682b053bab43

Doodle Hash output:



Figure 24: Generated Doodle Hash Sample 1

Input String 2:

d3490ff175dcb1779d09165827fb5b13bbfc179a

Doodle Hash output:



Figure 25: Generated Doodle Hash Sample 2

Input String 3:

fbcf35c06fe2e917951778e44eba603ca89206a9

Doodle Hash output:



Figure 26: Generated Doodle Hash Sample 3

Input String 4:

4f20cb8082e966987f45523a9d473ab862a913a0

Doodle Hash output:



Figure 27: Generated Doodle Hash Sample 4

Chapter 5 Usability Study & Design

In this section, we will be discussing the usability study and its design. The usability study is a survey that is designed to get feedback from people having different educational backgrounds and experiences. The feedback that we are aiming to acquire is regarding the time it takes for individuals to compare hash keys in their various forms, both visual and textual/character based. These various forms or schemes will help us study their effects on human performance, accuracy rate, and response times. For the collection of this data, we designed an online questionnaire in which the participants are asked to answer a series of comparison questions as well as their background/demographic information. The survey was shared amongst students and faculty of our university.

5.1 Design Goals

This study targets to answer the following queries[11]:

- 1) Does the selection of individuals in any way affect the accuracy or comparison times amongst the various hash schemes used in the survey?
- 2) Does the gender or age affect the accuracy or comparison response times amongst the various hash schemes used in the survey?
- 3) From the various schemes used in the survey, which one of those has a higher accuracy or comparison time?
- 4) Does the background knowledge of the respondents in any way affect the accuracy or comparison times?
- 5) Which scheme provides the highest accuracy?
- 6) Which scheme provides the quickest response time?

The survey also incorporates a division of the hash schemes based on the level of complexity of the pairs. There will be easy pairs and hard pairs in the survey with the likelihood of getting a hard pair being lower than the probability of getting an easy pair.

5.2 Designing Hash Pairs

To obtain fair results, all the different compression schemes should ideally contain the same level of information so that we can make a more accurate comparison amongst them. This is also known as entropy which is defined as bits per character. To achieve equal entropy amongst all the different hash schemes, we decided to first encrypt our information in a standard encryption protocol that will be used as a basis for all the various visualization schemes. We chose SHA-1 encryption as our base value which was then converted into the different visualization schemes.

Once the visual hash pairs were designed and stored into a pool of comparison images, each pair of visualization techniques was selected from that pool based on predefined probabilities. These selected images were then added to the questionnaire. The survey contained:

- pairs having identical representation have a probability of **0.5**.
- pairs having obviously different representations having a probability of **0.25**.
- pairs having very similar but different representations having a probability of **0.25**.

For each of the pairs presented to the respondents, they answered by pressing either the ‘same’ or ‘different’ button based on their perception[11]. The time taken by the respondents to answer each of the questions was also recorded and not shown to them. The image pairs generated were analyzed with a perceptual difference tool[34]. This tool takes two images as inputs and as an output, it displays whether the two images are the same or different[35]. Example Snapshots of PerceptualDiff tests performed are shown below:


```
C:\temp1>perceptualdiff C:\perc3\3a.png C:\perc3\3b.png -verbose -threshold 10 -output foo.ppm
Field of view is 45.000000 degrees
Threshold pixels is 10 pixels
The Gamma is 2.200000
The Display's luminance is 100.000000 candela per meter squared
PASS: Images are binary identical

C:\temp1>\
```

Figure 28: Perceptual Image Diff-Snapshot case1

```
C:\temp1>perceptualdiff C:\perc4\4a.png C:\perc4\4b.png -verbose -threshold 10 -output foo.ppm
Field of view is 45.000000 degrees
Threshold pixels is 10 pixels
The Gamma is 2.200000
The Display's luminance is 100.000000 candela per meter squared
Converting RGB to XYZ
Constructing Laplacian Pyramids
Performing test
FAIL: Images are visibly different
56571 pixels are different
```

Figure 29: Perceptual Image Diff-Snapshot case2

```
C:\temp1>start c:\temp1\perceptualdiff C:\ab\1a.png C:\ab\1b.png -verbose -threshold 10 -output foo.ppm
C:\temp1>perceptualdiff C:\perc2\1a.png C:\perc2\2a.png -verbose -threshold 10 -output foo.ppm
Field of view is 45.000000 degrees
Threshold pixels is 10 pixels
The Gamma is 2.200000
The Display's luminance is 100.000000 candela per meter squared
Converting RGB to XYZ
Constructing Laplacian Pyramids
Performing test
PASS: Images are perceptually indistinguishable
0 pixels are different
```

Figure 30: Perceptual Image Diff-Snapshot case3

5.3 Online Study

After the design and implementation of the usability study, the same usability study was used to perform an online survey to acquire and compare the time needed and accuracy for each of our selected hash comparison schemes. The survey form was designed on Microsoft Forms and the web link of the survey was shared with faculty members and students at our university. When participants opened our survey, they completed two key steps: fill in their background data and perform 24 hash

comparisons. In the first step, contributors were first requested to mention their email, name, age group, gender, educational background, and their experience and familiarity with doodle hash. At the end of the survey, the participants were asked to rate the six schemes based on their level of ease and usability. In the second step, after the collection of the demographics, the users compared 24 pairs of visual hash representations. The participants were instructed to visually compare two illustrations of a single scheme at a time and choose whether the two representations were the identical or not. The selection process of hash schemes has been discussed earlier. For each question prompt, the time is taken by the participant from the moment the question was viewed till the time the participant selected an answer was stored by the form along with the answer selected by the participant. In the survey, the same and different images of each scheme, having one of three difficulty levels (easy, medium, hard), were shown to the participants.

The study was divided into four forms (A, B, C, D) to randomize the order of comparison questions. So, the users were able to answer and perform the comparisons in different order of schemes. Snapshots of usability study questionnaires are shared below.

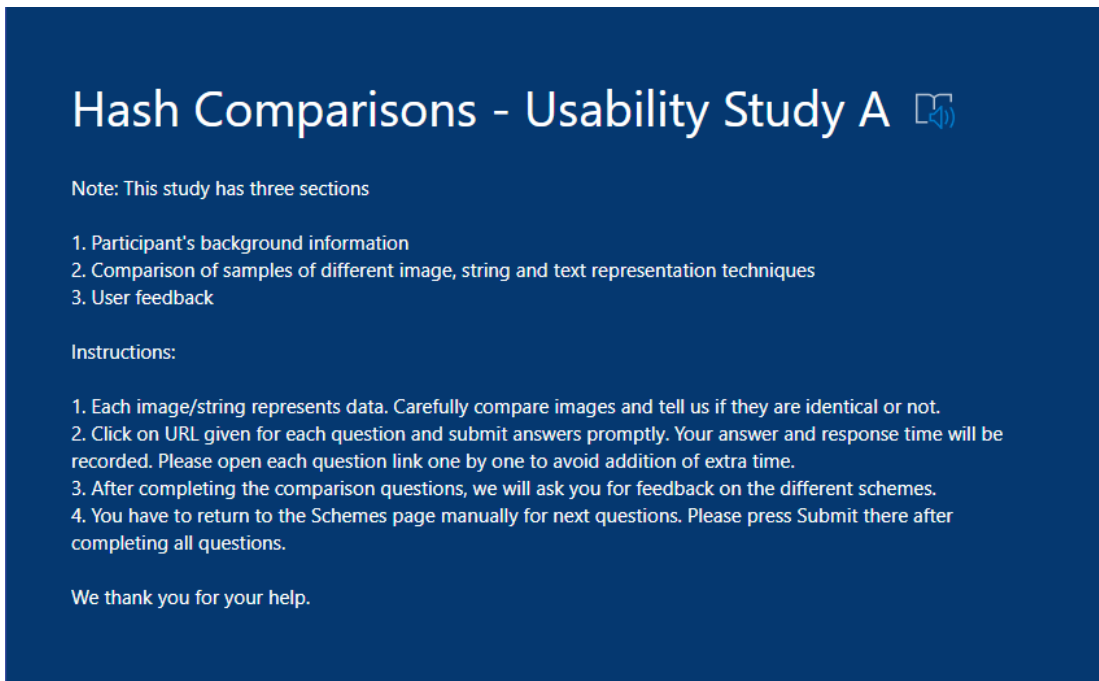


Figure 31: Usability Study Snapshot 1

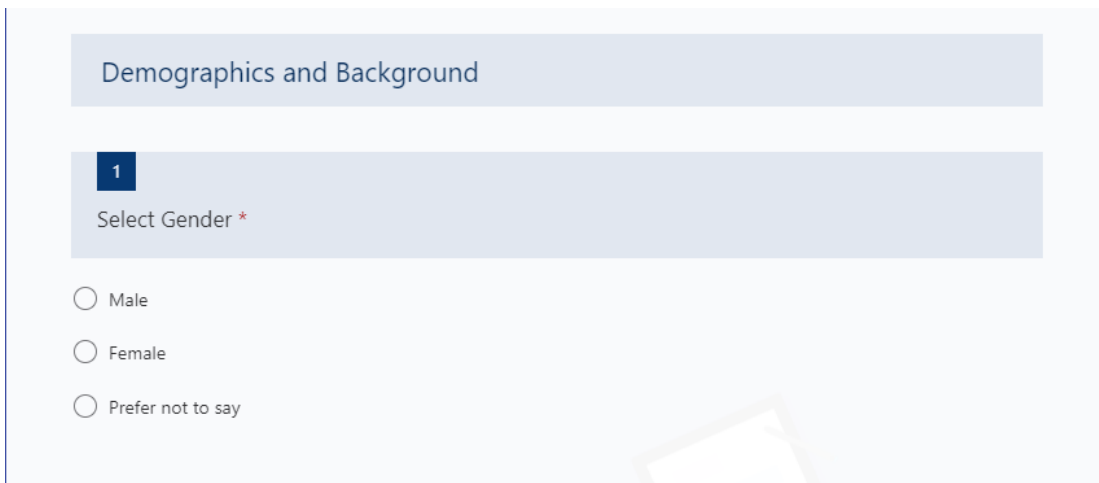
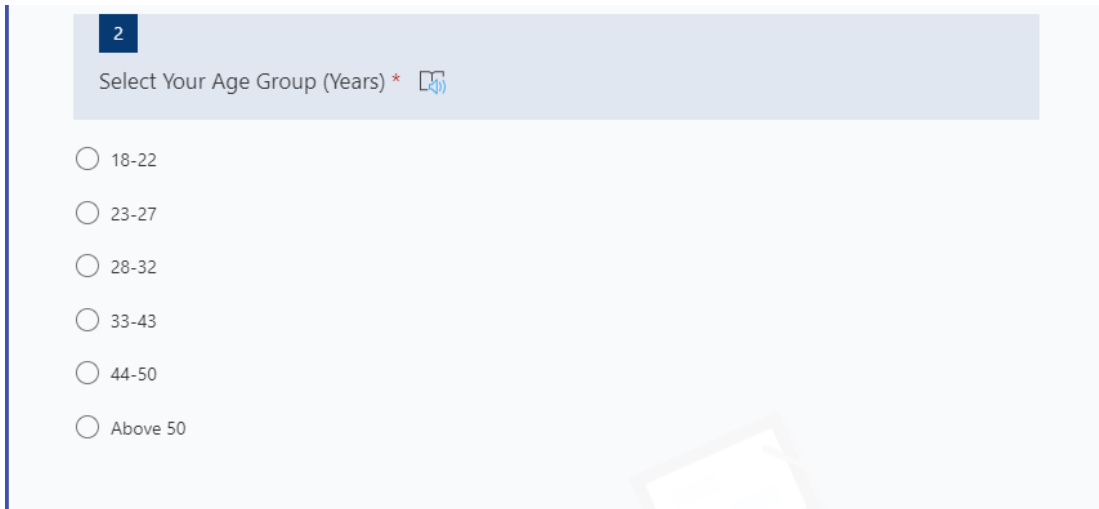



Figure 32: Usability Study Snapshot 2

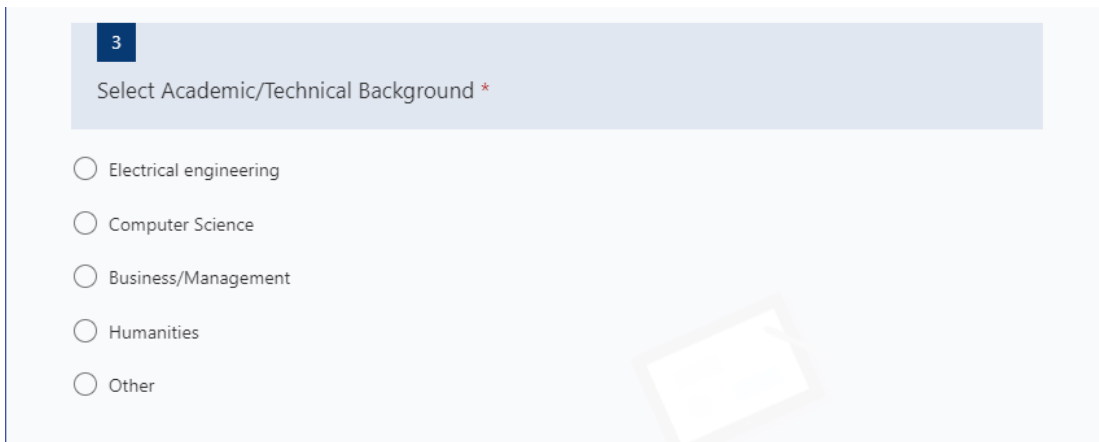


2

Select Your Age Group (Years) * 

- 18-22
- 23-27
- 28-32
- 33-43
- 44-50
- Above 50

Figure 33: Usability Study Snapshot 3



3

Select Academic/Technical Background *

- Electrical engineering
- Computer Science
- Business/Management
- Humanities
- Other

Figure 34: Usability Study Snapshot 4

4

Have you any prior experience with using cryptographic credentials or hash functions? *

Yes

No

5

Are you familiar with use of doodles as a technique to represent and compare data? *

Yes

No

Figure 35: Usability Study Snapshot 5

Some snapshots of Hash Visualization Schemes comparison questions

1

Are these two images identical or different? * 



Identical

Different

Submit

Figure 36: Hash Comparison Snapshot 1-TFlag

1

Are these two strings identical or different? *

```
HE4TEYTCGUZDSYRYMQZWIZTBGQ3DEOLDMI2DOOLFMYZTCNDDMM3DINJXMRRTONRT
```

```
HE4TEYTCGUZGEOJYMQZWIZTBGQ3DEOLDMI2DOOLFMYZTCNDDMM2DMNJXMRRTONRT
```

Identical

Different

Submit

Figure 37: Hash Comparison Snapshot 2-Base32

1

Are these two strings identical or different? *

```
'robert-tango-stream-sierra'
```

```
'robert-tango-steam- sierra'
```

Identical

Different

Submit

Figure 38: Hash Comparison Snapshot 3-Alpha Hash

1

Are these two strings identical or different? *

3c1TKDhoNLYhbDqxyEgLwbcKn2BuJzD8nhYWJQ5xGRk4C6+TW2gTgV5

3c1TKDhoNLYhbDqxyEgLwbcKn2BuJzD8nhYWJQ5xGRk4C6+TW2gTgV5


Identical


Different

Submit

Figure 39: Hash Comparison Snapshot 4-Base58

1

Are these two images identical or different? * 



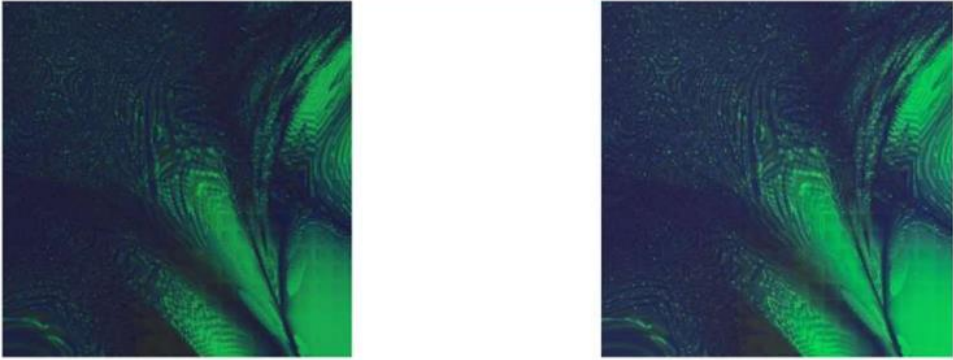
Identical

Different

Figure 40: Hash Comparison Snapshot 5-Doodle Hash

1

Are these two images identical or different? *




Identical

Different

Submit

Figure 41: Hash Comparison Snapshot 6-Random Art

User Feedback and Rating Question:

User Feedback 

Instructions:

We request you evaluate all schemes in terms of

1. Ease of use
2. Aesthetics
3. Personal satisfaction

Figure 42: Usability Study Snapshot 6

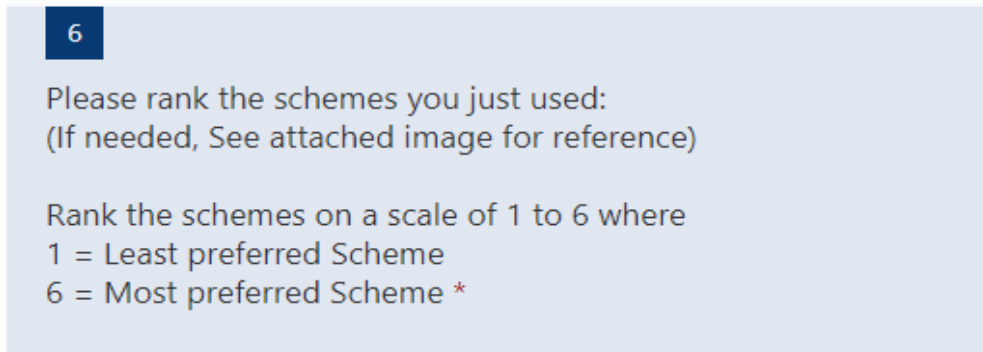


Figure 43: Usability Study Snapshot 7

	1	2	3	4	5	6
Base32	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Random Art	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Alpha Hash	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Doodle Hash	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Base58	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
TFlag	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Figure 44: Usability Study Snapshot 8

This is how the schemes rating were acquired from each participant after performing the comparison questions.

5.4 Participant Demographics

Our survey was shared amongst the faculty and students at our university. There were total of 89 male and 42 female contributors. The age group of the users with respect to their counts as 57 belonging to the 18-22 age group, 58 belonging to the 23-27 age group, 13 belonging to the 28-32 age group, and 3 belonging to the 33-43 age group. 50% of the participants belonged to the computer science department, 31% belonged to the electrical engineering department, and 19% belonged to other departments (Students of Mechanical engineering, natural sciences, and Biosciences). Out of all the participants, only 16% were familiar with the use of hash schemes.

Chapter 6 Results and Analysis

6.1 Data Preprocessing:

Upon collecting and analyzing the participants' feedback, we performed data cleaning by identifying outliers using the $1.5 \times \text{IQR}$ method. As an example, the response times for the Base-32 easy comparison, the maximum response time was 180 seconds. After applying the $1.5 \times \text{IQR}$ rule, we eliminated the outliers, and then the maximum response time that we got was approximately 100 seconds.

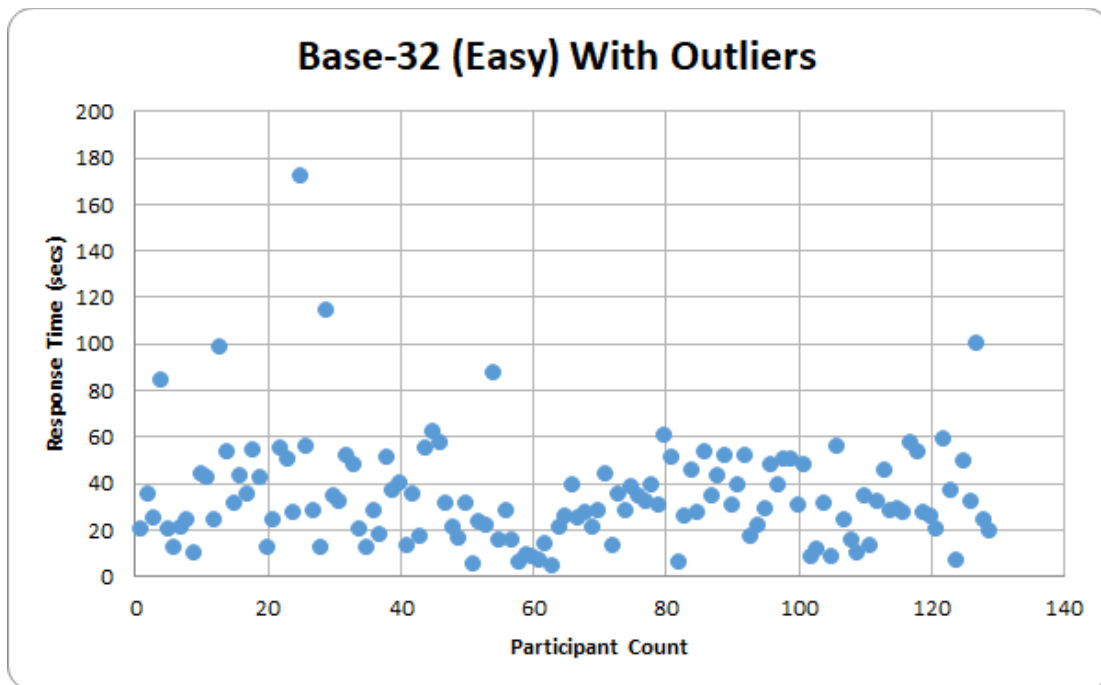


Figure 45: Base-32 (Easy) with outliers

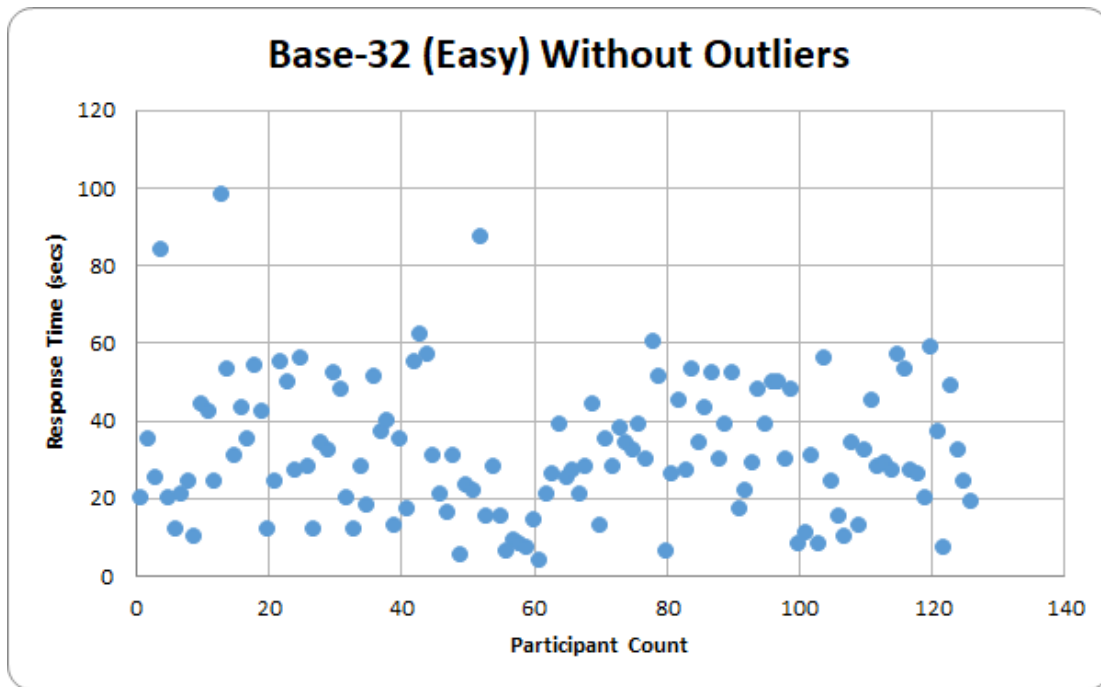


Figure 46: Base-32 (Easy) without outliers

6.2 Results and Analysis

In this section, the results of the online study are presented and analyzed. To make a fair comparison between the different hash schemes, selective parameters are taken into consideration from the gathered results. The accuracy and response times for all three types of questions (easy, medium, and hard) are analyzed for different age groups, gender, and hash comparison schemes.

6.2.1 Factor wise impact

6.2.1.1 Gender:

The percentage of participation in the survey by gender was 66% males and 34% females. On average, the response time of males was more than females by 2 seconds.

Count of Participants by Gender

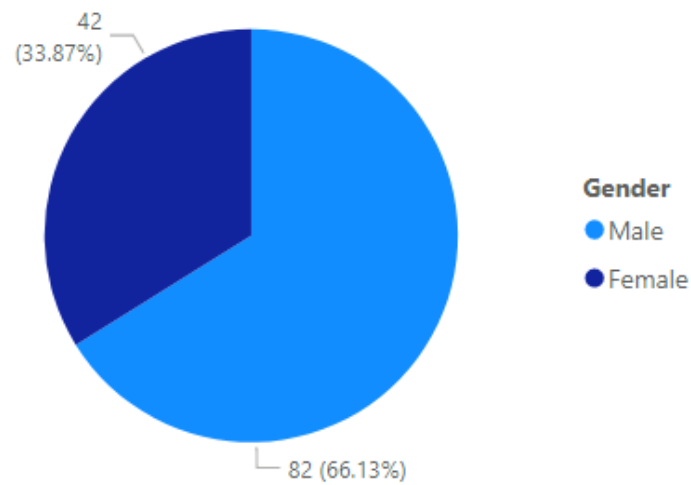


Figure 47: Count of Participants by Gender

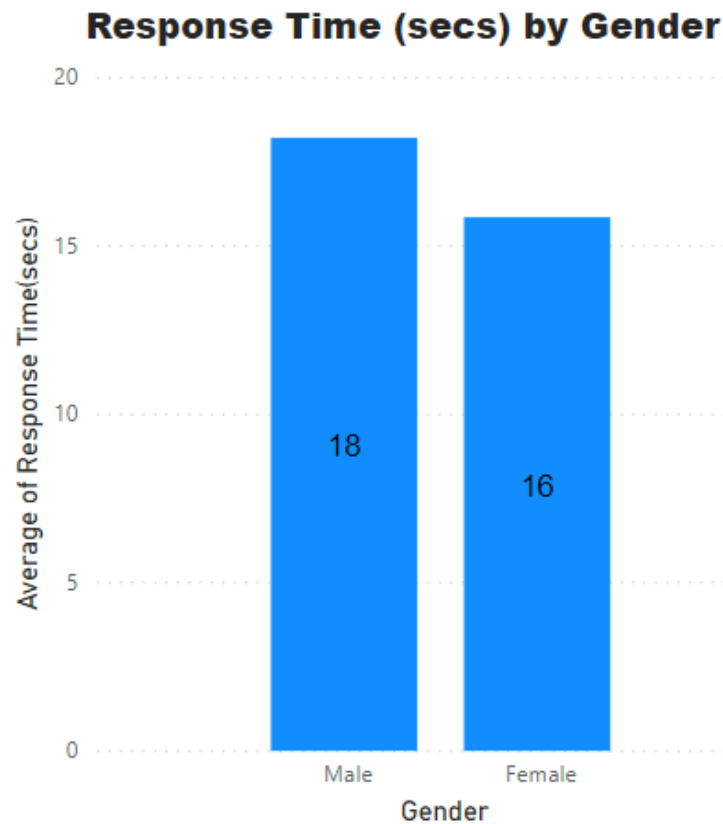


Figure 48: Response Time (secs) by Gender

6.2.1.2 Age Group:

The age groups based on the average of all the response times sorted in descending order are 23-27, 18-22, 33-43, and 28-32. The majority of the participants belonged to the 23-27 age group while the age group 33-43 had only three participants.

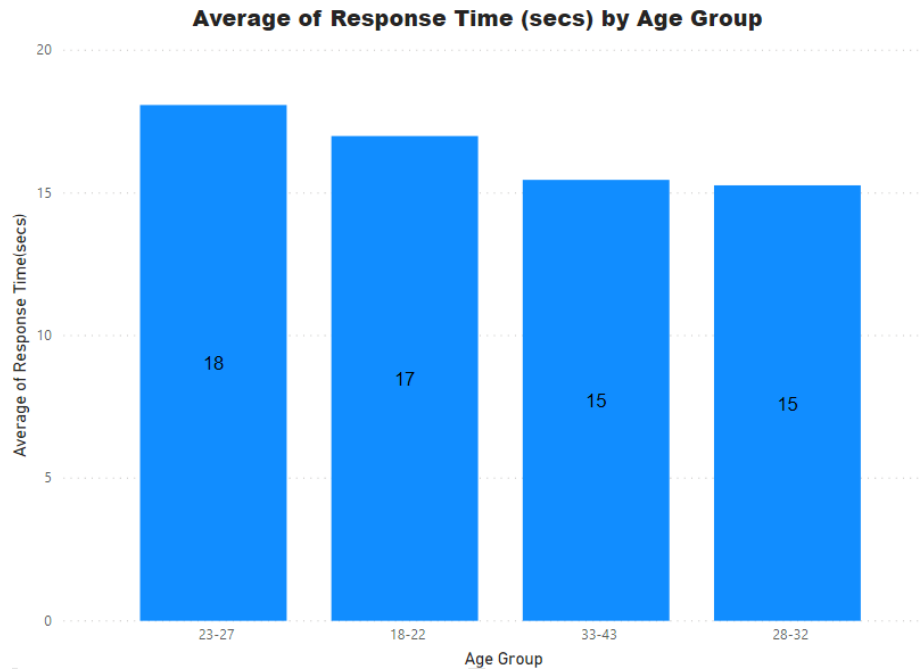


Figure 49: Average of Response Time (secs) by Age Group

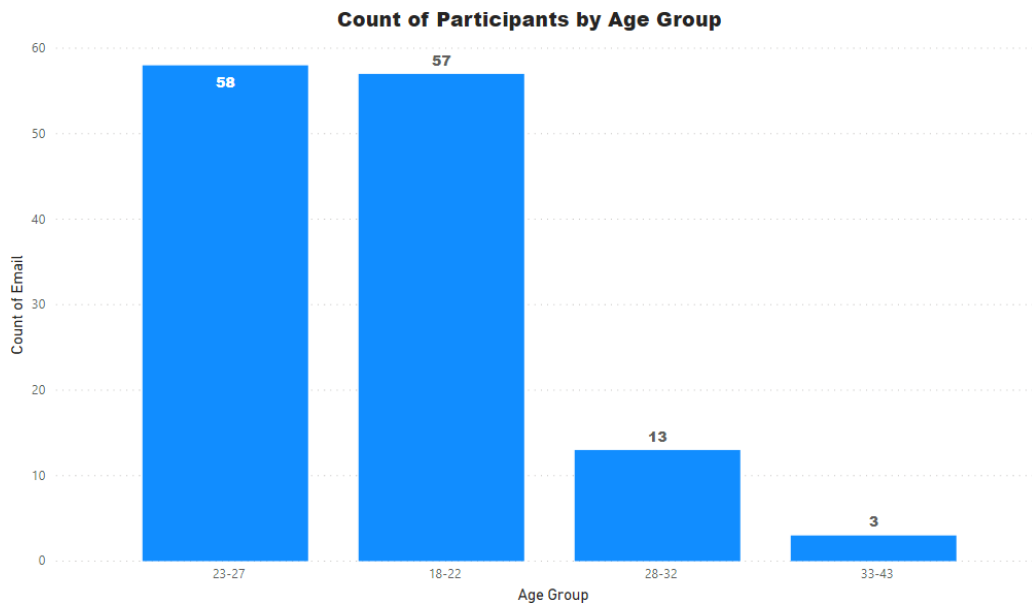


Figure 50: Count of Participants by Age Group

6.2.1.3 Background:

The background information of all the participants was also collected in the usability study. Out of all the participants, 50% were from the computer science background,

31% had a background in Electrical Engineering, and 19 percent were from other educational backgrounds.

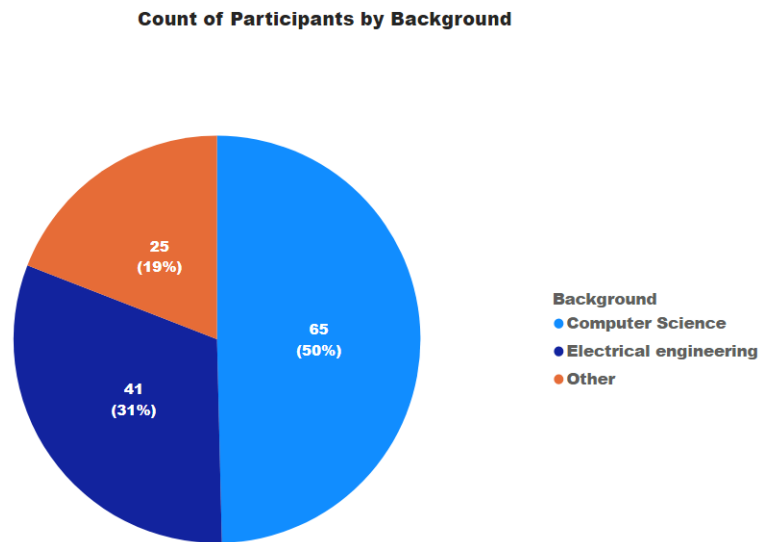


Figure 51: Count of Participants by Background

Upon comparing the background of the participants with their respective response times, we can see that those having a computer science background took the most time on average in responding to the survey questions followed by those having an electrical engineering background. Participants having other backgrounds showed the least response time on average.

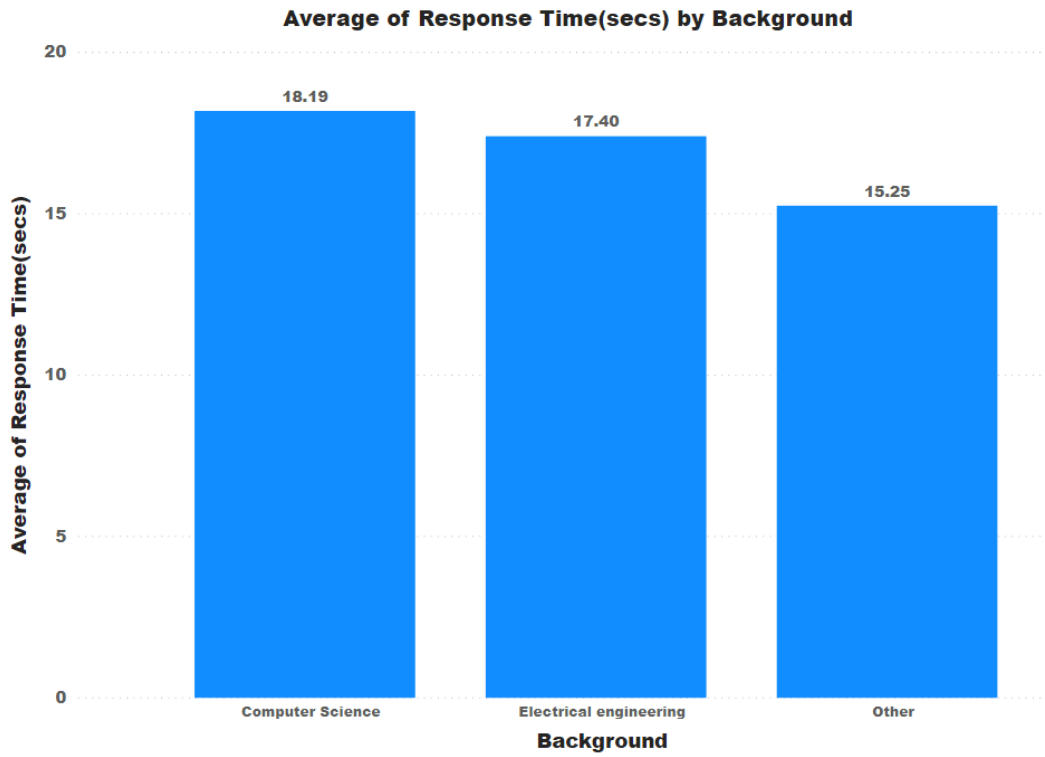


Figure 52: Average of Response time by Background

6.2.1.4 Familiarity:

Out of all the participants, 16% were familiar with doodles while the rest of the 84% had no such familiarity.

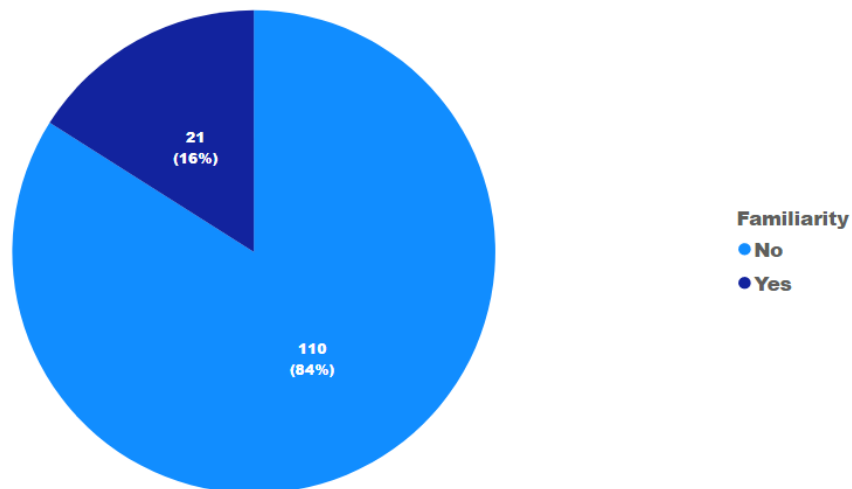


Figure 53: Count of Participants by Familiarity

It can be seen from the figure below that those who were familiar with doodles took more time on average than those who were not familiar with doodles.

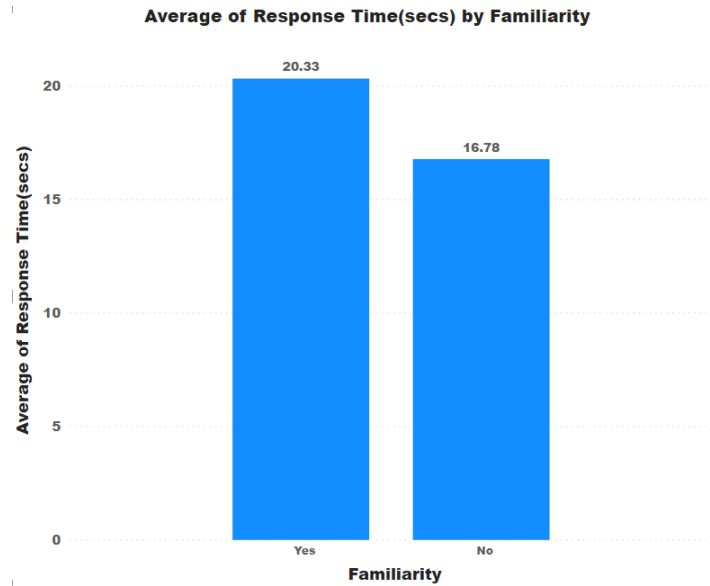


Figure 54: Average of Response Time by Familiarity

6.2.1.5 Experience:

When asked from the participants whether they had any prior experience relating to the use of cryptographic functions or hash keys, only 23% stated that they had some sort of prior experience while the rest of the 77% of the participants stated that they had no such knowledge.

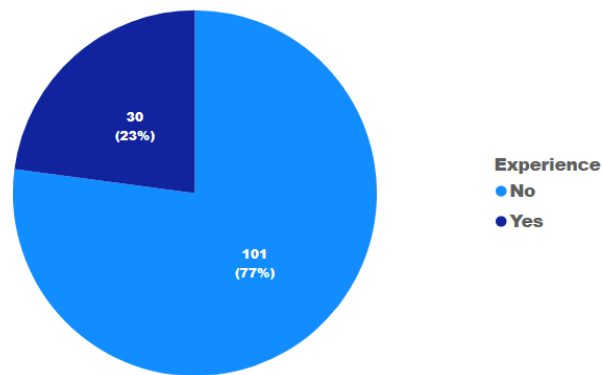


Figure 55: Count of Participants by Experience

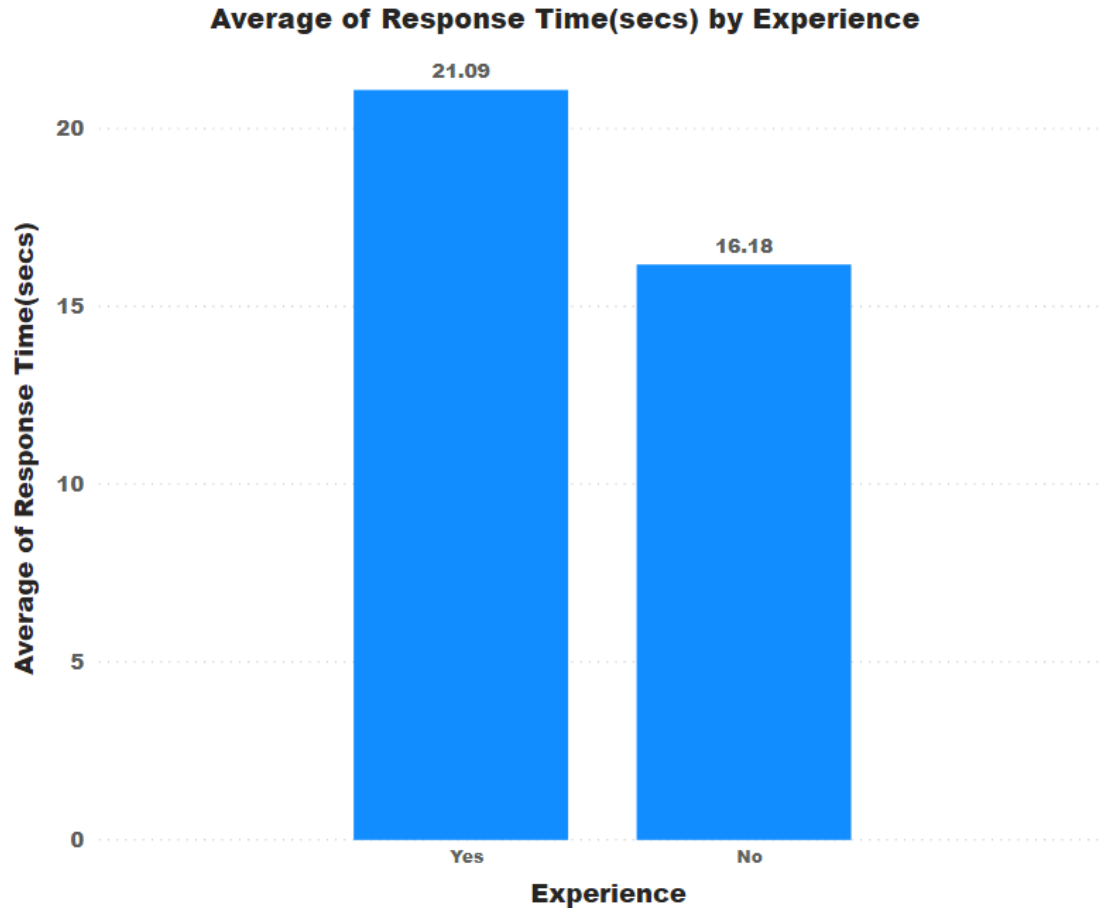


Figure 56: Average of Response Time by Experience

From the figure above, we can see that the participants that had some from of cryptographic experience, on average they took more time to response to the survey questions.

6.2.2 Performance of Each Hash Comparison Scheme

6.2.2.1 Alpha Hash

In alpha hash, we saw that the highest accuracy was of the easy pair and the lowest accuracy was of the hard pair. The highest average response time was of the easy pair and the lowest response time was of the medium pair.

Level	Average Time (secs)	Minimum Time (secs)	Maximum Time (secs)	Accuracy
Easy	13.7	3	52	100%
Medium	11.4	2	58	97%
Hard	11.6	1	71	67%

Table 1: Alpha Hash Summary

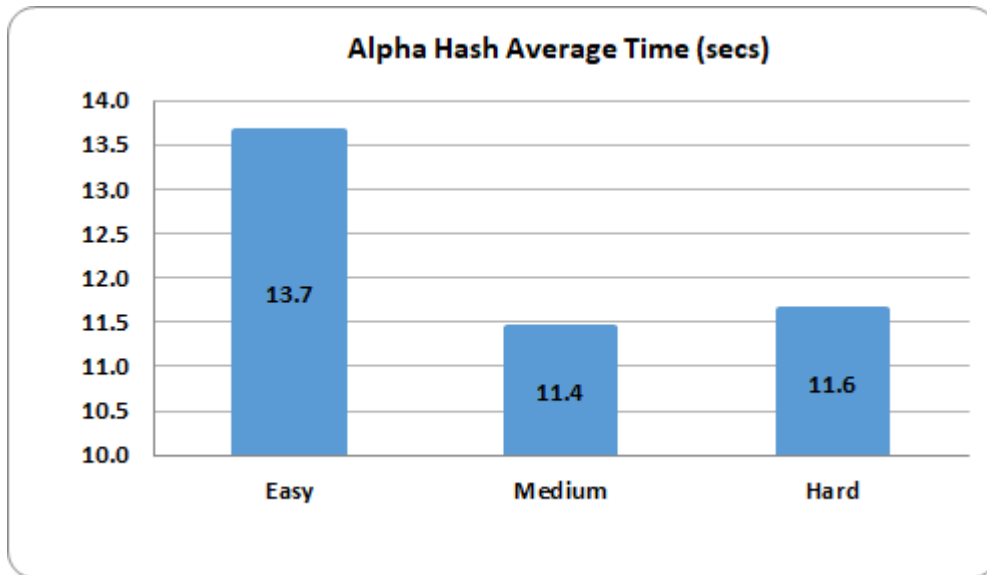


Figure 57: Alpha Hash Response Time

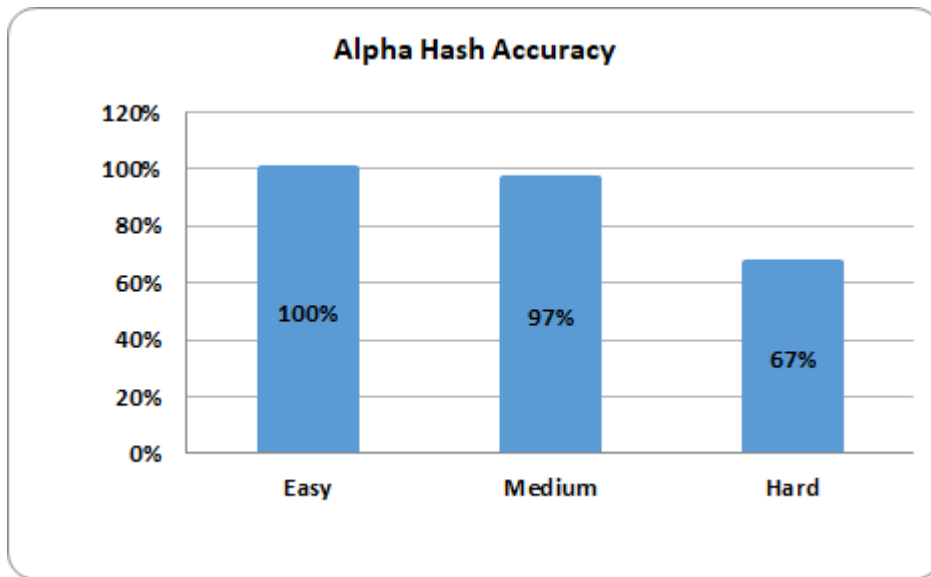


Figure 58: Alpha Hash Accuracy

6.2.2.2 Base-32

In base-32 we saw that the highest response time was for the easy pair while the lowest response time was for the medium pair. Similarly, the highest accuracy was of the easy pair while the lowest accuracy was of the hard pair.

Level	Average Time (secs)	Minimum Time (secs)	Maximum Time (secs)	Accuracy
Easy	30.7	3	95	99%
Medium	12.7	2	99	97%
Hard	18.6	3	84	85%

Table 2: Base32 Summary

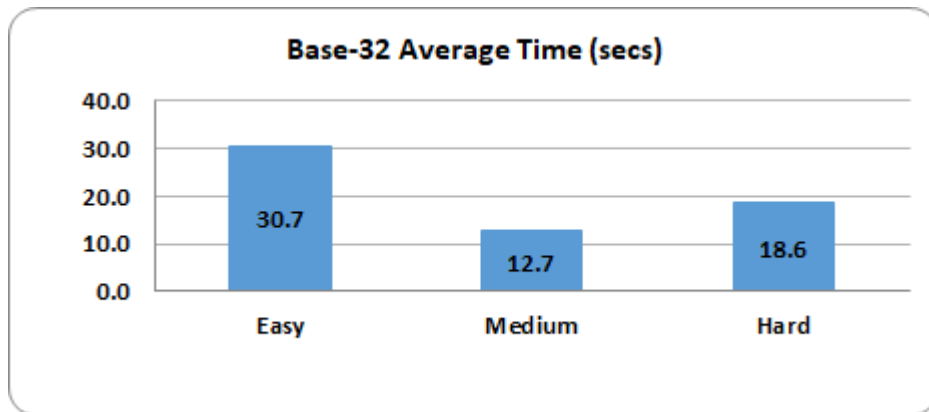


Figure 59: Base-32 Average Time (secs)

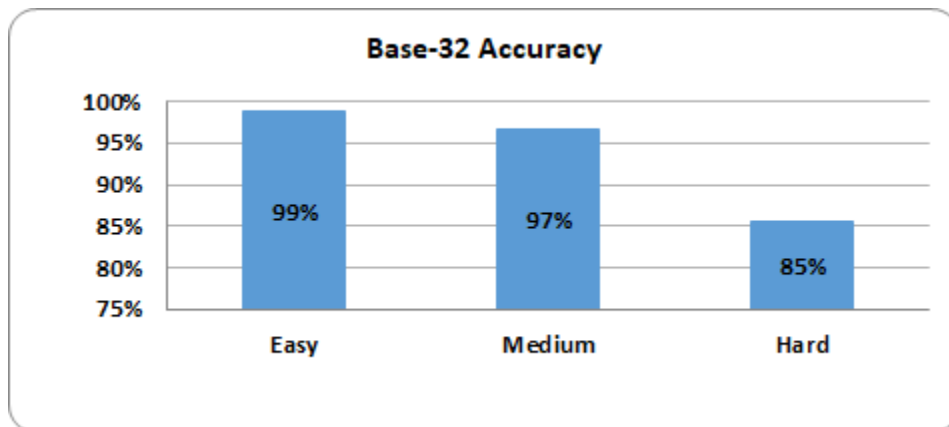


Figure 60: Base-32 Accuracy

6.2.2.3 Base-58

For base-58 we saw that the highest average time was of the easy pair while the lowest average time was of the medium pair. Similarly, the highest accuracy was of the easy pair while the lowest accuracy was of the hard pair.

Level	Average Time (secs)	Minimum Time (secs)	Maximum Time (secs)	Accuracy
Easy	24.9	2.5	72.5	99%
Medium	12.8	1.0	78.0	97%
Hard	21.9	4.0	88.0	64%

Table 3: Base58 Summary

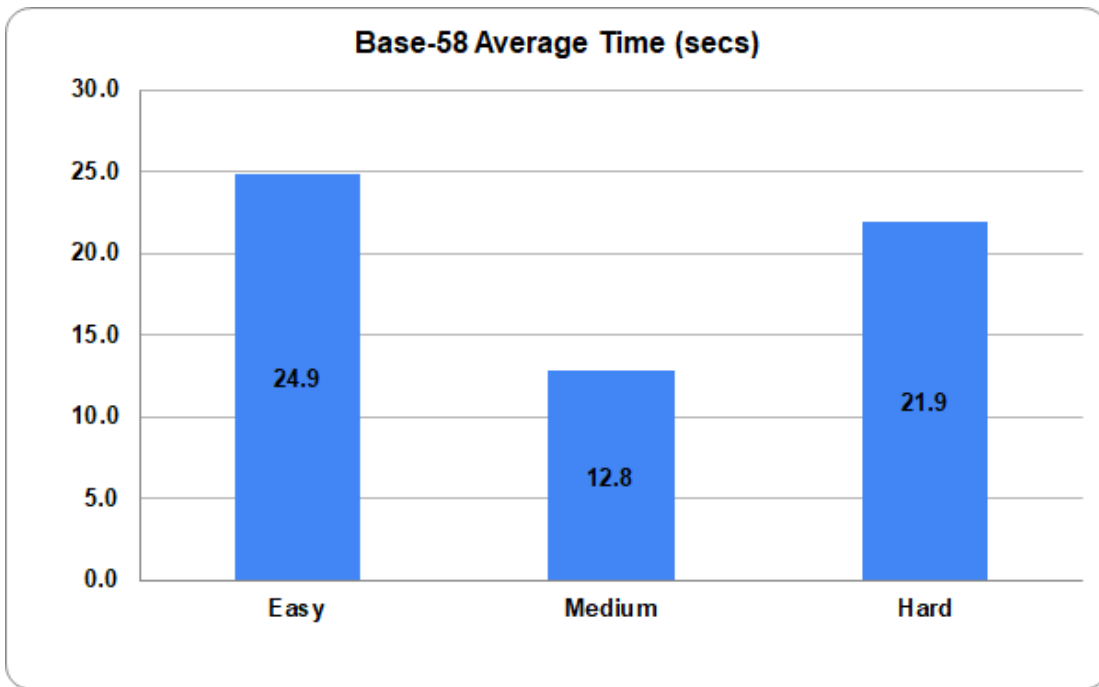


Figure 61: Base-58 Average Time(secs)

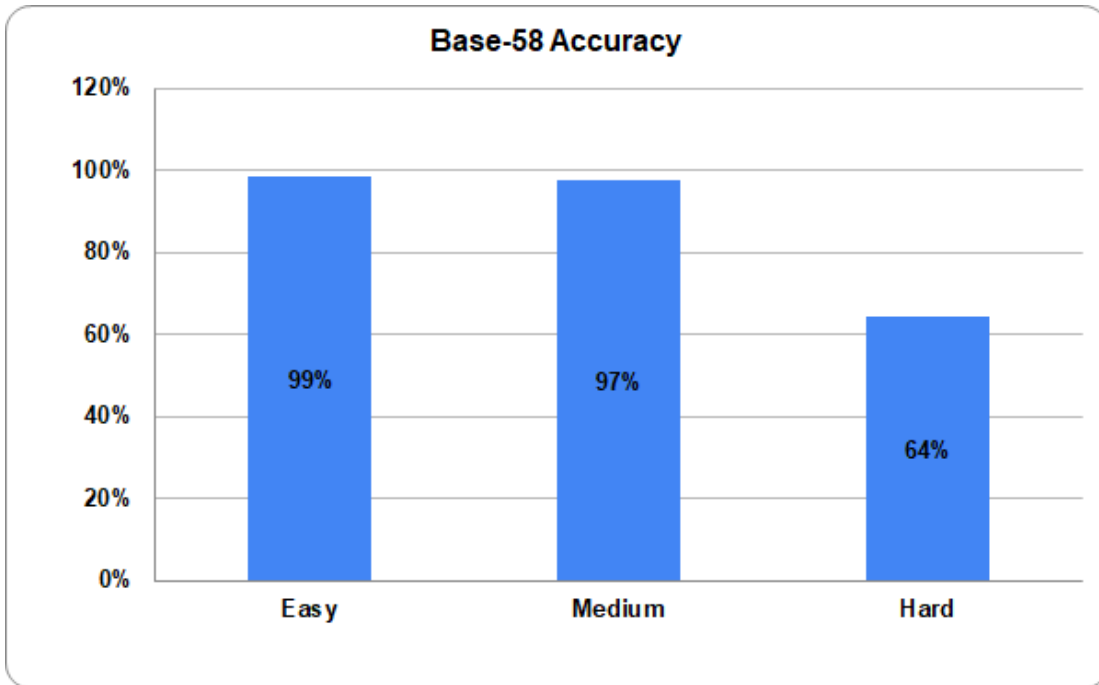


Figure 62: Base-58 Accuracy

6.2.2.4 Random Art

For Random art, we saw that the highest response time was of both the easy and hard pairs while the lowest response time was of the medium pair. Similarly, the highest accuracy was of the medium pair while the lowest accuracy was of the hard pair.

Level	Average Time (secs)	Minimum Time (secs)	Maximum Time (secs)	Accuracy
Easy	19	2	86	82%
Medium	11	2	94	100%
Hard	19	3	78	68%

Table 4: Random Art Summary

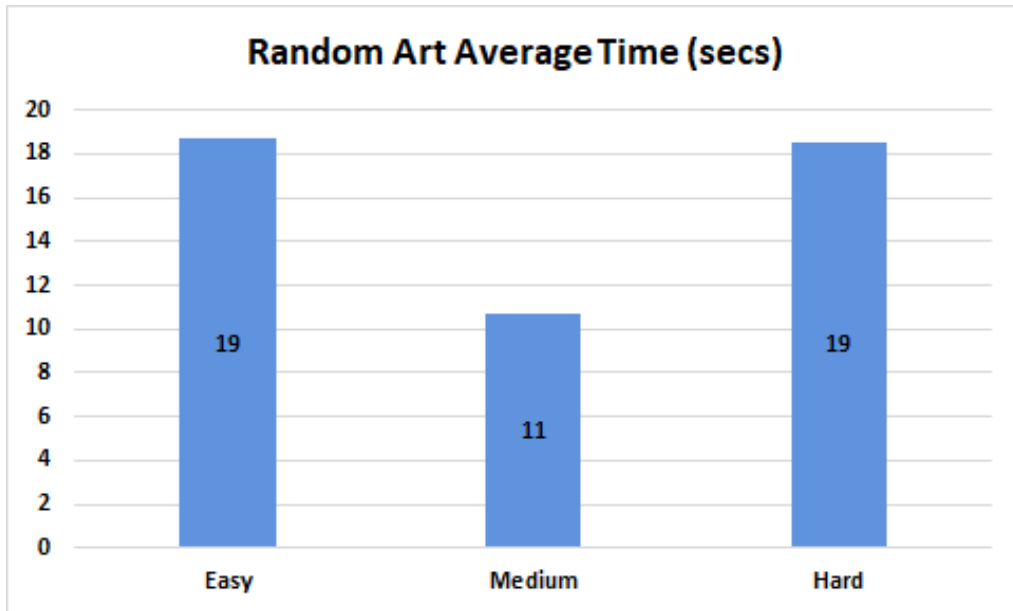


Figure 63: Random Art Average Time(secs)

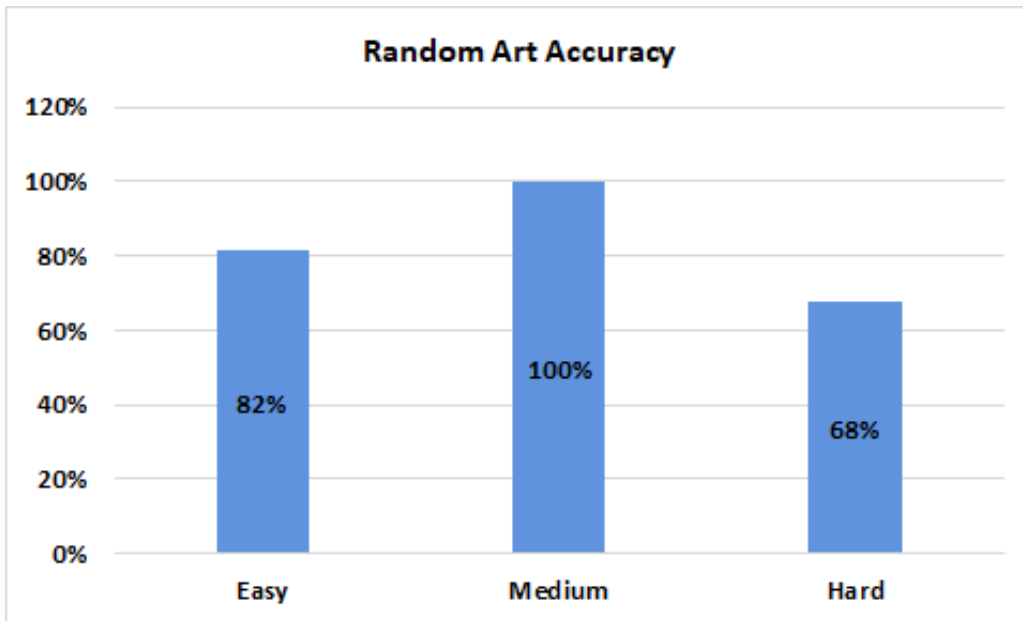


Figure 64: Random Art Accuracy

6.2.2.5 T-Flag

For T-Flag, we saw that the highest response time was of the easy and hard pairs while the lowest response time was of the medium pair. Similarly, the highest accuracy was of the medium pairs while the lowest accuracy was of the hard pairs.

Level	Average Time (secs)	Minimum Time (secs)	Maximum Time (secs)	Accuracy
Easy	13	2	70	97%
Medium	12	2	72	98%
Hard	13	2	64	52%

Table 5: TFlag Summary

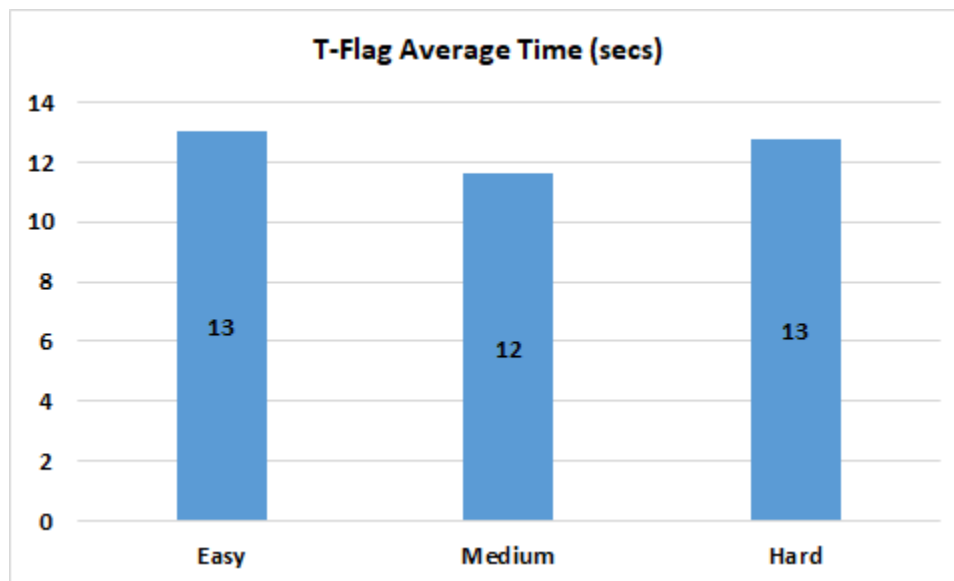


Figure 65: T-Flag Average Time

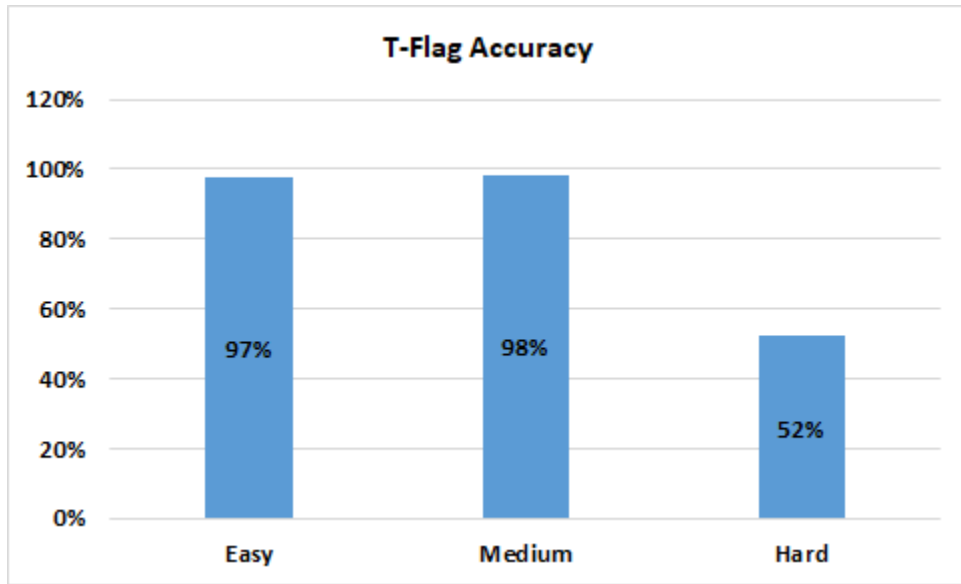


Figure 66: T-Flag Accuracy

6.2.2.6 Doodle Hash

For doodle hash, we saw that the highest response time was of the easy pair while the lowest response time was of the medium pair. Similarly, the highest accuracy was of the medium pair while the lowest accuracy was of the hard pair.

Level	Average Time (secs)	Minimum Time (secs)	Maximum Time (secs)	Accuracy
Easy	31	3	330	98%
Medium	20	3	217	99%
Hard	27	4	310	69%

Table 6: Doodle Hash Summary

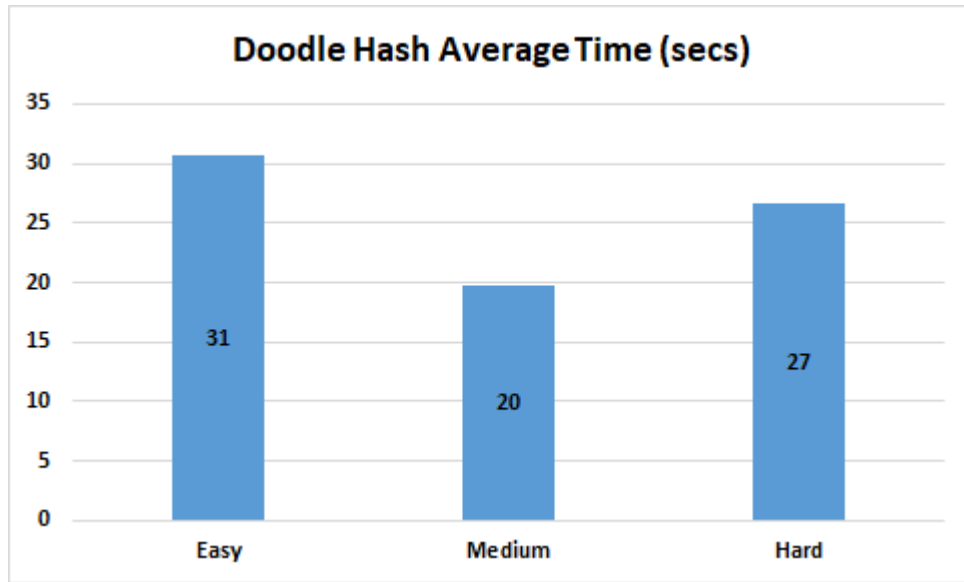


Figure 67: Doodle Hash Average Time(secs)

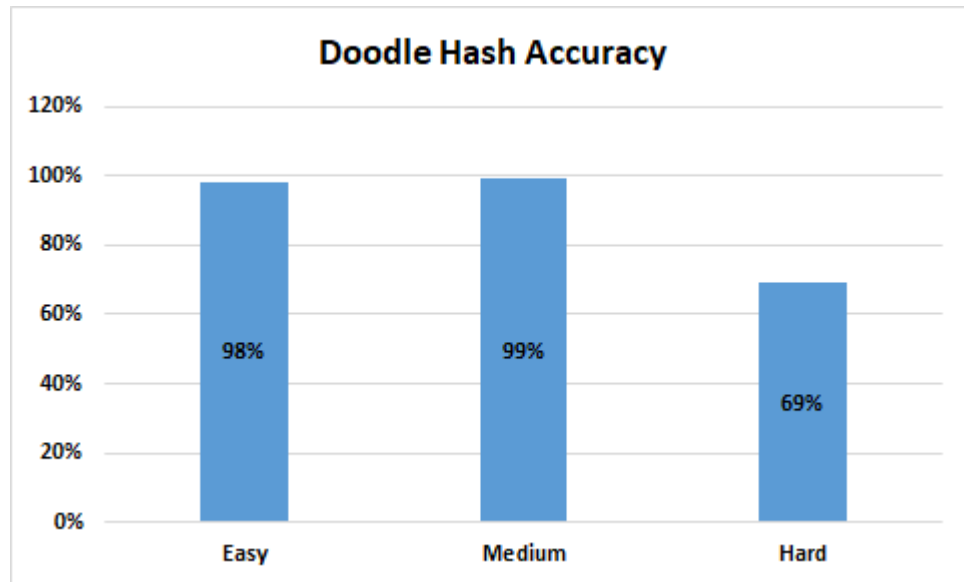


Figure 68: Doodle Hash Accuracy

Chapter 7 Conclusion and Future Work

7.1 Additional Properties of Hash Comparison Schemes

Talking more about the hash schemes that have been taken into account for the purpose of this research, in order to be used for the purpose of cryptography, various properties of the hash schemes have to be taken into account in order to make a balanced conclusion as to which scheme or more are preferred for the purpose of cryptography. Out of the six schemes discussed, two of them require to be compared on the basis of colors. This can be troublesome for many as the limitations of being colorblind can hinder the process of cryptographic key comparisons.

Scheme	Achromatic	Multicolored
Alpha Hash	✓	
Base-32	✓	
Base-58	✓	
Random Art		✓
T-Flag		✓
Doodle Hash	✓	

Table 7: Additional properties of schemes

7.2 Discussion

Upon combining the obtained results, we aggregated the response times and accuracies for the different schemes. Looking at the graph showing the aggregated response times for all the schemes, at a first glance, we can see that the lowest average response time obtained from the participants is of the alpha hash scheme and the highest average response time is of base-32 scheme. One thing to be noted here is that as the focus of this research is to identify the best hash scheme possible in terms of accuracy and response time, therefore the levels of the comparison pairs have been aggregated scheme wise.

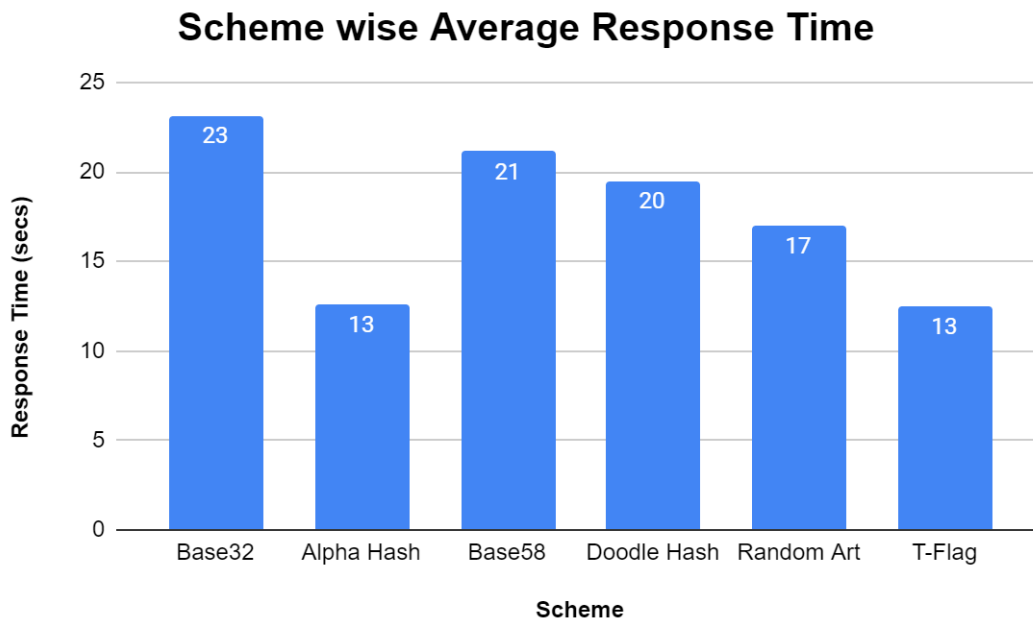


Figure 69: Scheme wise Average Response Time

Similarly, for the graph showing the scheme wise average accuracy, we can see that the highest accuracy is of the base-32 scheme and the lowest accuracy is of the random art scheme. Once again, the different levels of all the schemes have been aggregated by taking averages so that we may focus our results and findings to the schemes only.

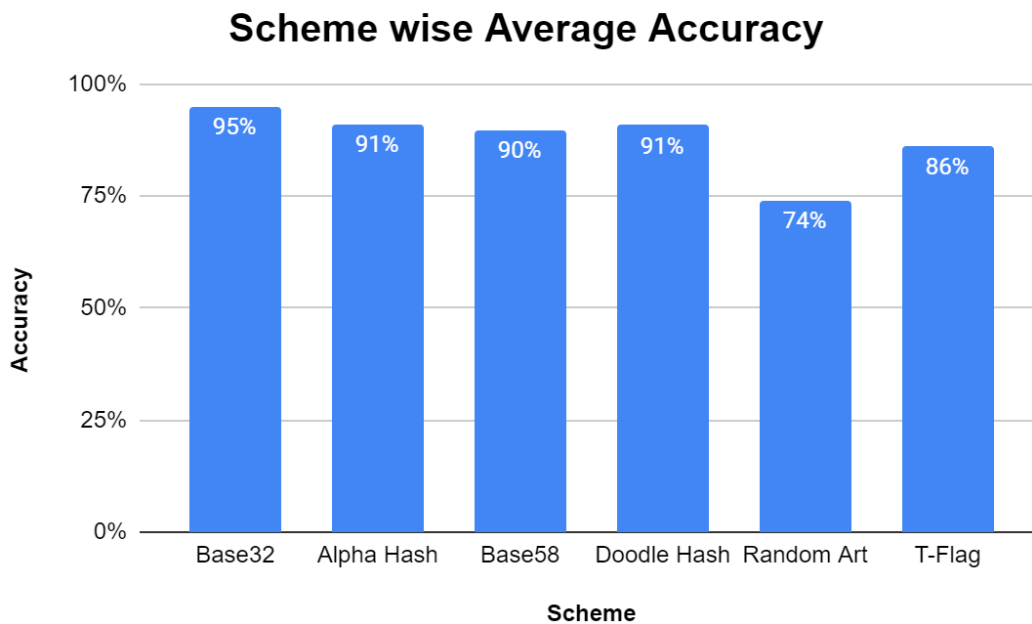


Figure 70: Scheme wise Average Accuracy

In order to make a justified comparison between the six schemes, both the accuracy and response times of all the schemes have to be analyzed together because one scheme may have the lowest response time, making it a very quick-to-use scheme but at the same time, the same scheme might have a very low accuracy. The figure below shows both these factors compounded onto one graph. The ideal scheme will have the highest accuracy and lowest response time.

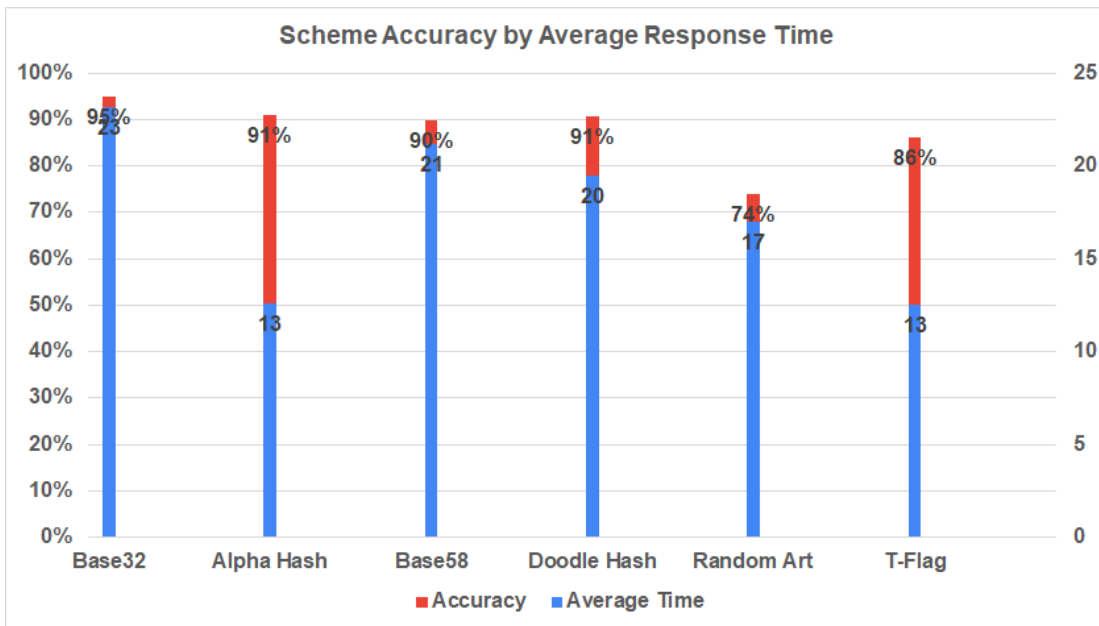


Figure 71: Scheme Accuracy by Average Response Time

Looking at the results in the graph above, we see that base-32 has the highest accuracy but it also has the highest response time, making it unsuitable in comparison to the other schemes. Random art and t-flag have accuracies that are lower than the rest so we ignore them for now. The performance of doodle hash is better than base-58 by having a higher accuracy and a lower response time. Alpha hash has the best accuracy to response time ratio out of all the other schemes. An important thing to be highlighted here is that in the survey, for the case of alpha hash, the participants had to only compare sets of four words each but in the case of doodle hash, the participants had to compare doodles sets of 20 doodles. This deems it to be obvious that the participants would take less time to compare the 4 words in alpha hash in contrast to the 20 doodles in doodle hash.

7.3 Conclusion

To make a fair comparison among the various schemes, we decided to develop a ratio that can help us in making a justified conclusion and rank the schemes in terms of both accuracy and response time. The ratio that we have used is calculated by dividing the

accuracy by response time for each of the schemes. The higher the ratio, the better the performance of the scheme. The results have been ranked in the table below.

Rank	Scheme	Accuracy	Response Time (secs)	Ratio
1	Alpha Hash	91%	13	0.0724
2	T-Flag	86%	13	0.0688
3	Doodle Hash	91%	20	0.0465
4	Random Art	74%	17	0.0435
5	Base58	90%	21	0.0424
6	Base32	95%	23	0.0410

Table 8: Summary of Hash schemes

From the ranking above, we can see that Doodle hash is placed 3rd out of the six schemes. But if we make a comparison between achromatic schemes, then the doodle hash scheme is placed second, below the alpha hash.

After conducting the usability study with more than 100 participants, we analyzed the performance of existing hash schemes both alphanumeric (Base32, Base58), Graphical schemes (TFlag, Random Art), Alpha Hash (English Words) with each other and with our proposed scheme Doodles Hash.

We conclude that our proposed scheme can be used as a valid alternative to long alphanumeric hash strings and make a good addition in Graphical Visualization schemes. Doodle Hash provides significant accuracy with less response time than other schemes like Base32 and Base58. The response time of TFlag and Random Art is lower

than doodle Hash but there is also a compromise on Accuracy values as shown in table above. Hence, we can show that Doodle Hash provides good results when compared to other schemes. Also Doodle Hash does not depend on color display and resolution display settings of the device which is a highly impacting factor in other visual schemes under consideration i.e., Random Art and TFlag which represent different data with changed visual display. There is no such dependency in the Doodle Hash scheme that may alter the represented hashed data.

Based on the obtained results after performing the usability study analysis, we present that the Doodle Hash scheme has preference over the traditional strings and other visual schemes as it provides high accuracy with reduced response time. Doodle Hash is suggested to use as a Hash visualization technique as it provides the good tradeoffs among accuracy, response time and usability.

7.4 Future Work

The Doodle Hash scheme can be used in various applications for example as an authentication tool, validations of keys, e-voting receipts verification, Bitcoin transaction validations and many other platforms where the traditional alphanumeric strings are used. In future, the doodle hash system is open for development and testing using different doodles datasets that can generate variants of doodle hash strings and may provide different results in terms of accuracy, response time and usability.

Bibliography

- [1] “What is hashing?” <https://www.educative.io/edpresso/what-is-hashing> (accessed May 22, 2021).
- [2] “Integrity - Cybersecurity Glossary.” <https://cybersecurityglossary.com/integrity/> (accessed May 22, 2021).
- [3] “SHA1 online.” <http://www.sha1-online.com/> (accessed May 22, 2021).
- [4] S. Xiaoyuan, Z. Ying, and G. S. Owen, “Graphical passwords: A survey,” *Proc. - Annu. Comput. Secur. Appl. Conf. ACSAC*, vol. 2005, no. Acsac, pp. 463–472, 2005, doi: 10.1109/CSAC.2005.27.
- [5] O. Lobachev, “Direct visualization of cryptographic keys for enhanced security,” *Vis. Comput.*, vol. 34, no. 12, pp. 1749–1759, 2018, doi: 10.1007/s00371-017-1466-6.
- [6] J. Tan, L. Bauer, J. Bonneau, L. F. Cranor, J. Thomas, and B. Ur, “Can unicorns help users compare crypto key fingerprints?,” *Conf. Hum. Factors Comput. Syst. - Proc.*, vol. 2017-May, pp. 3787–3798, 2017, doi: 10.1145/3025453.3025733.
- [7] A. Perrig, A. Perrigscmuedu, D. Song, and D. Songscmuedu, “Hash Visualization : a New Technique to improve Real-World Security 1 Introduction 2 Requirements for Hash Visual- ization Algorithms,” *Int. Work. Cryptogr. Tech. E-Commerce*, pp. 1–8, 1999.
- [8] S. Dechand, D. Schürmann, K. Busse, Y. Acar, S. Fahl, and M. Smith, “An empirical study of textual key-fingerprint representations,” *Proc. 25th USENIX Secur. Symp.*, pp. 193–208, 2016.
- [9] “The Effect of Background Luminance and Contrast upon Visual Search Performance | Building Research Information Knowledgebase.” <https://www.brikbases.org/content/effect-background-luminance-and-contrast->

- upon-visual-search-performance (accessed May 22, 2021).
- [10] A. Garba, Z. Guan, A. Li, and Z. Chen, “Analysis of man-in-the-middle of attack on bitcoin address,” *ICETE 2018 - Proc. 15th Int. Jt. Conf. E-bus. Telecommun.*, vol. 2, no. Icete, pp. 388–395, 2018, doi: 10.5220/0006864003880395.
- [11] H. C. Hsiao *et al.*, “A study of user-friendly hash comparison schemes,” *Proc. - Annu. Comput. Secur. Appl. Conf. ACSAC*, pp. 105–114, 2009, doi: 10.1109/ACSAC.2009.20.
- [12] R. S. A. Daulay, S. M. Nasution, and M. W. Paryasto, “Realization and Addressing Analysis in Blockchain Bitcoin,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 260, no. 1, 2017, doi: 10.1088/1757-899X/260/1/012002.
- [13] M. R. R. Fauzi, S. M. Nasution, and M. W. Paryasto, “Implementation and Analysis of the use of the Blockchain Transactions on the Workings of the Bitcoin,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 260, no. 1, 2017, doi: 10.1088/1757-899X/260/1/012003.
- [14] M. Brengel and C. Rossow, *Identifying key leakage of bitcoin users*, vol. 11050 LNCS, no. February 2014. Springer International Publishing, 2018.
- [15] S. Zhai, Y. Yang, J. Li, C. Qiu, and J. Zhao, “Research on the Application of Cryptography on the Blockchain,” *J. Phys. Conf. Ser.*, vol. 1168, no. 3, 2019, doi: 10.1088/1742-6596/1168/3/032077.
- [16] Y. H. Lin *et al.*, “SPATE: Small-group PKI-less authenticated trust establishment,” *IEEE Trans. Mob. Comput.*, vol. 9, no. 12, pp. 1666–1681, 2010, doi: 10.1109/TMC.2010.150.
- [17] N. S. Govindarajulu and S. Madhvanath, “Password management using doodles,” *Proc. 9th Int. Conf. Multimodal Interfaces, ICMI’07*, pp. 236–239, 2007, doi: 10.1145/1322192.1322233.
- [18] M. Martinez-diaz, J. Fierrez, and J. Galbally, “Graphical Password-Based User Authentication With Free-Form Doodles,” vol. 46, no. 4, pp. 607–614, 2016.

- [19] N. S. Govindarajulu and S. Madhvanath, “Doodles for Authentication : Recognition and User Study Results,” *Methods*, 2008.
- [20] J. Goldberg, J. Hagman, and V. Sazawal, “Doodling our way to better authentication,” *Conf. Hum. Factors Comput. Syst. - Proc.*, pp. 868–869, 2002, doi: 10.1145/506621.506639.
- [21] M. Martinez-Diaz, J. Fierrez, C. Martin-Diaz, and J. Ortega-Garcia, “DooDB: A graphical password database containing doodles and pseudo-signatures,” *Proc. - 12th Int. Conf. Front. Handwrit. Recognition, ICFHR 2010*, pp. 339–344, 2010, doi: 10.1109/ICFHR.2010.59.
- [22] C. Varenhorst and others, “Passdoodles: A lightweight authentication method,” *Res. Sci. Inst.*, pp. 1–13, 2004, [Online]. Available: http://people.csail.mit.edu/emax/public_html/papers/sow-2004/varenhorst.pdf.
- [23] M. Azimpourkivi, U. Topkara, and B. Carbunar, “Human distinguishable visual key fingerprints,” *Proc. 29th USENIX Secur. Symp.*, pp. 2237–2254, 2020.
- [24] M. Maina Olembo, T. Kilian, S. Stockhardt, A. Hülsing, and M. Volkamer, “Developing and testing SCoP - A visual hash scheme,” *Inf. Manag. Comput. Secur.*, vol. 22, no. 4, pp. 382–392, 2014, doi: 10.1108/IMCS-11-2013-0082.
- [25] “Base58. Bitcoin’s underlying technology is... | by Paige Cabianca | nakamo.to | Medium.” <https://medium.com/nakamo-to/what-is-base58-c6c2db7808f3> (accessed May 22, 2021).
- [26] “Ben Nuttall - MD5 Flag Generator.” <https://bennuttall.com/md5-flag-generator/> (accessed May 22, 2021).
- [27] “Random art: gallery.” <http://www.random-art.org/> (accessed May 22, 2021).
- [28] “Random art: about random art.” <http://www.random-art.org/about/> (accessed May 22, 2021).
- [29] “humanhash3 · PyPI.” <https://pypi.org/project/humanhash3/> (accessed May 22,

- 2021).
- [30] “quickdraw-dataset/categories.txt at master · googlecreativelab/quickdraw-dataset · GitHub.” <https://github.com/googlecreativelab/quickdraw-dataset/blob/master/categories.txt> (accessed May 22, 2021).
 - [31] Kristine Guo, James WoMa, and Eric Xu, “Quick, Draw! Doodle Recognition Challenge,” *Kaggle*, no. 345, 2018.
 - [32] “GitHub - googlecreativelab/quickdraw-dataset: Documentation on how to access and use the Quick, Draw! Dataset.” <https://github.com/googlecreativelab/quickdraw-dataset> (accessed May 22, 2021).
 - [33] “PyFPDF.” <https://pyfpdf.readthedocs.io/en/latest/> (accessed May 22, 2021).
 - [34] “Perceptual Image Difference Utility.” <http://pdiff.sourceforge.net/> (accessed May 23, 2021).
 - [35] “Using the Perceptual Difference (Pdiff) Tool | www.seleniumrecipes.com.” <http://www.seleniumrecipes.com/content/using-perceptual-difference-pdiff-tool> (accessed May 23, 2021).