DETECTING WEB APPLICATION'S BROKEN AUTHENTICATION BY USING COMBINATORIAL ALGORITHM

OHNMAR THET

M.C.Sc. **JANUARY 2023**

DETECTING WEB APPLICATION'S BROKEN AUTHENTICATION BY USING COMBINATORIAL ALGORITHM

By

Ohnmar Thet

B.C.Sc.

A dissertation submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science (M.C.Sc.)

University of Computer Studies, Yangon

JANUARY 2023

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my sincere thanks to those who helped me with various aspects of conducting research and writing this thesis. To complete this thesis, many things are needed like my hard work as well as the supporting of many people.

First and foremost, I would like to express my deepest gratitude and my thanks to **Dr. Mie Mie Khin**, Rector, the University of Computer Studies, Yangon, for her kind permission to submit this thesis.

I would like to express my appreciation to **Dr. Si Si Mar Win and Dr. Tin Zar Thaw**, Professors, Faculty of Computer Science, University of Computer Studies, Yangon, for their superior suggestion, administrative supports and encouragement during my academic study.

My thanks and regards go to my supervisor, **Dr. Zin Thu Thu Myint, Associate Professor**, Faculty of Information Science, University of Computer Studies, Yangon, for her support, guidance, supervision, patience and encouragement during the period of study towards completion of this thesis.

I also wish to express my deepest gratitude to **Daw Aye Aye Khine**, Associate Professor, Department of English, the University of Computer Studies, Yangon, for her editing this thesis from the language point of view.

Moreover, I would like to extend my thanks to all my teachers who taught me throughout the master's degree course and my friends for their cooperation.

Last but not least, I especially thank to my parents, all of my colleagues, and friends for their encouragement and help during my thesis.

ABSTRACT

Authentication and session management functions in web applications are used to verify the identity of the user. Incorrect implementation of these functions allows attackers to compromise passwords, keys or sessions tokens. When a user is authenticated, an authenticated session can be established which usually gives the user increased usability of the application, such as access to the user's private data. The purpose of this system is to conduct a web application security using simple brute force and dictionary attack in broken authentication with combinatorial algorithm. There is no best way to protect the user's computer security, but always try to improve upon what it has. The evaluation is implemented with the line graph of the time consuming and password length. This system is implemented using C# programming language with Microsoft SQL Server Database Engine.

Key Word: Authentication, brute force, dictionary attack, combinatorial algorithm

CONTENTS

ACKNOWLEDGEMENTS ..... **i**
ABSTRACT ..... **ii**
CONTENTS ..... **iii**
LIST OF FIGURES ..... **v**
LIST OF TABLES ..... **vi**
LIST OF EQUATIONS ..... **vii**
CHAPTER 1 ..... **1**
INTRODUCTION **1**
1.1Objectives of the Thesis 1
1.2 Related Works 2
1.3 Organization of the Thesis 3
CHAPTER 2 ..... **4**
BACKGROUND THEORY **4**
2.1 Top 10 Open Web Application Security Project (OWASP) Attacks 4
2.1.2 Injection 4
2.1.2 Broken Authentication and Session Management 5
2.1.3 Cross-Site Scripting (XSS) 6
2.1.5 Security Misconfiguration 7
2.1.6 Sensitive Data Exposure 8
2.1.7 Missing Function Level Access Control 9
2.1.8 Cross-Site Request Forgery 9
2.1.9 Using Components with Know Vulnerabilities 11
2.1.10 Un-validated Redirects and Forward 12
2.2 Brute Force Attack 12
2.3 Types of Brute Force Attacks 14
2.4 Tools Aid Brute Force Attempts 15
2.4.1 GPU Speeds Brute Force Attempts 16
2.5 Steps to Protect Passwords for Professionals 16
2.6 How Users Can Strengthen Passwords Against Brute Force Attacks 18
2.7 Dictionary Based Attack 19
2.8 Work of Dictionary Based Attack 20
2.9 Brute-force attack vs. dictionary attack 20
2.10 How to protect yourself against a dictionary attack 22

CHAPTER 3 ..... **23**
DESIGN OF THE PROPOSED SYSTEM **23**
3.1 Combination Algorithm for Brute Force Attack 24
3.2 Approximate Time, Combination and Password Length for Brute Force 26
3.**3** Dictionary for Attack 27
3.**4** The System Flow 29
3.**5** The Evaluation of the Proposed System 30
CHAPTER 4 ..... **31**
IMPLEMENTATION OF THE PROPOSED SYSTEM **31**
4.1. Experimental Setup 31
4.2 Implementation of System Design 31
4.3 Experiment Results 37
CHAPTER 5 ..... **39**
CONCLUSION, LIMITATIONS AND FURTHER EXTENSIONS **39**
5.1 Advantages 39
5.2 Limitations and Further Extensions 39
REFERENCES 40

LIST OF FIGURES

Page

Figure 2.1 2020 Passwords table use data from 2018 based on MD5 hashed passwords cracker by an RTX 2080 GPU (HIVE systems) 13

Figure 2.2 Brute Force Attack 21

Figure 2.3 Dictionary Attack 22

Figure3.1 Sample Password Combination of Brute Force Attack 25

Figure 3.2 Sample Password Contents of Dictionary Based Attack 28

Figure 3.3 The System Flow 29

Figure 4.1 Main Form of the proposed system 31

Figure4.2 Status for Starting Dictionary Attack 32

Figure 4.3 The Status for Dictionary password generating success 33

Figure 4.4 The status box for success dictionary attack with guess password 33

Figure 4.5 The status for Brute Force generation process 34

Figure 4.6 Evaluation Results for Dictionary Based Attacks (1) 35

Figure 4.7 Evaluation Results for Dictionary Based Attacks (2) 35

Figure 4.8 Evaluation Results for Dictionary Based Attacks (3) 36

Figure 4.9 Login page of the target online book shopping web application 36

Figure 4.10 Main page of the online book shopping web application 37

Figure 4.11 Line graph for Brute Force and Dictionary Based Attack crack time 38

LIST OF TABLES

Page

Table 3.1 Approximate time for brute force based on the character length 26

Table 4.1 Estimated Cracking Time based on the password length 38

LIST OF EQUATIONS

Page

Equation 3.1 Combination Equation 24

Equation 3.2 Equation for possible combinations for all chosen characters 26

CHAPTER 1
INTRODUCTION

In the act of using web applications, the sensitive personal information that people give are stored on these web applications.On the other hand, some unethical attackers exploit the web application to gain unauthorized access and illegal use of personal information and do other things such as identity theft, privacy violation, and other cyber-attacks. These illegal points allow the attackers to make whatever they want through the weaknesses of the web application. The weak point of the web application called vulnerability caused by unawareness of the developers who cannot be controlled validation the user inputs, appropriate validation methods, and so on. Because of those facts, detection of vulnerability is needed more.

This system will check whether the system provided a secure login function or not and to know how secure the authentication is. And it will give a message on whether or not the authentication is secured. In this system, the first step is to guess the password of the login page from the proposed web application. This system will guess the password in two ways: the dictionary based and brute-force attacks. In a dictionary attack, the system will use a wordlist library of common passwords/pass-phrases, add user-specific words (social engineered information), add “pass phrases” to the wordlists, generate variants for each word using several common “complexification” patterns. In dictionary attack, conduct the password of login page with the dictionary word in word file that is pre-created and pass the system via the success guess password. For brute force attack, the system will use target language letter frequency combinations, use a wordlist (letters/digits/symbols) as the building blocks of component, try to crack a passphrase (the sequence of work). In brute-force attack, generating the pattern using combination algorithm and guess the possible password to pass through the login page of web page. The outcome of the security testing will be displayed as a graph that were noticed during the detection.

1.1Objectives of the Thesis

The main objectives of this thesis are:

- to examine whether login password of website is vulnerable or not
- to examine whether the website login password is weak enough to crack

- to check how the website login is weak enough to attack by brute force
- to check how the website login is weak enough to attack by dictionary based
- to study the security nature of authentication

1.2 Related Works

As a first study showed that Brute force attacks are used to login to network services with pairs of username and passwords. For network service administrators, brute force attacks are the main security issues. The network administrators have to set the limits of login attempts and block the traffic of brute force attacks with an intrusion prevention system (IPS) at the entry point. In recent years, stealthy brute force attacks that can overcome the security rules and IPS and intrusion detection system (IDS) detection have shown up [9]. This paper, also give the description of a kind of distributed brute force attack event against the Remote Desktop Protocol (RDP) by identification IDS logs integrated from multiple sites. DBF can make a repeated particular number of attacks automatically from a host to a service over a period of time. For this reason, existing countermeasures have no issue on DBF. Inspecting the structure of DBF will enhance the existing countermeasure system. Presenting TOPASE, which is restored at each step of the existing countermeasure system and is relevant for DBF countermeasures. TOPASE inspect the regularity of login trials between a source host and a destination host and intercepts the network traffic from the source host of the brute force attack for a specific period time. As a result of the evaluation, it can estimate the performance of TOPASE and analyze the reasons that expand TOPASE's strength.

A common threat Brute force attacks are a common threat in contrast to web applications and may have issue in compromising user accounts, which lead to unauthorized access to user data and transactions. Signal Sciences facilitate DevOps and security teams to encounter and safeguard against brute force attacks with Power Rules. Brute Force Attacks are tried to guess the username and password to obtain illegitimate access to the application functionality. There are many different shapes and sizes in Brute Force:

- Guessing username and passwords of the target
- web application authentication
- Lists of user profiles

- Catalogs of web server directories

Some web application securities are caused by the human factor such as the lack of knowledge of password security and strong password creation policies on them. Furthermore, letting users to create the common password can also increase the weakness of the system. [9] This study used the passwords from the students of Slovenian university to access the online grading system by combining several well-established cracking attacks such as brute-force, dictionary and hybrid attacks. The aim was to expose how easy it is to crack most of the user-created passwords using simple and predictable password patterns. To measure the strength of password, it takes an analysis of the cracked and uncracked passwords. The results have shown that even a single low to mid-range modern GPU can crack over 95% of passwords in just few days, while a more dedicated system can crack all but the strongest 0.5% of them. [10]

1.3 Organization of the Thesis

The thesis is organized in five chapters. They are as follows:

In Chapter 1, introduction of the system, objectives of the thesis, related works and thesis organization are described. Chapter 2 presents the background theory. Chapter 3 discusses the design of the proposed system. And Chapter 4 expresses the implementation of the proposed system. Finally, Chapter 5 presents the conclusions, advantages of system and limitations and further extension of the system.

CHAPTER 2
BACKGROUND THEORY

In this chapter, the related background theory is presented. The top ten attacks of the Open Web Application Security Project (OWASP), the benefit for the attackers, types of brute force attacks, how to protect passwords, how to strengthen password against Brute force attacks, the dictionary attacks are described as a unique specific section.

2.1 Top 10 Open Web Application Security Project (OWASP) Attacks

The Open Web Application Security Project, or OWASP [8] is the nonprofit organization focus on the web application security. The OWASP Top Ten is a regularly updated description that identifies the ten most significant risks to web application security.

2.1.2 Injection

An attacker uses injection to embed (or inject) their own code into software by taking advantage of insecure code. Attackers are able to use injection attacks to gain access to secure data and private information as though they are trusted users because the program is unable to analyze the code embedded in this manner from its own code. The examples of injection are SQL injections, command injections, CRLF injections, and LDAP injections.

- **SQL injection:** Malicious code is inserted into statements via SQL injection, through the input of a web page. The attacker writes any file to the server and even execute OS commands by using SQL commands as SQL Injection attacks. This could compromise the entire system.
- **Command injection:** The attacker inserts operating system commands with the acknowledgement of the web application's user. In more advanced scenarios, the attacker may take advantage of additional authority escalation bugs, which could result in the complete system compromise.
- **CRLF infusion:** An unexpected collection of CRLF characters is injected by the attacker. CRLF infusion can be accomplished by splitting an HTTP response header

and editing arbitrary content to the response body. Cross-site scripting (XSS) may be used in association with this attack.

- **LDAP infusion:** To carry out any LDAP commands, the intruder inserts LDAP (Lightweight Directory Access Protocol) statements. They have the ability to access permissions and modify the LDAP tree's contents.
- **SMTP Header Injection:** This attack is related to CRLF injections. A web application cannot directly access because the intruder delivers IMAP/SMTP commands to a mail server.
- **Injection of the host header:** Utilizing the implicit trust of the HTTP Host header, the attacker poisons password-rest functionality and web caches.
- **Injection of OS commands:** The attacker injects operating system commands using the permissions granted by the user of the web application. The attackers may exploit additional privilege escalation flaws in more advanced scenarios, which could compromise the entire system.
- **Injection of XPath:** The intruder inserts data into an application in order to execute crafted XPath queries. They can make use of them to circumvent authentication and gain access to data that is not theirs.

2.1.2 Broken Authentication and Session Management

An OWASP-indexed vulnerability that acknowledges the risk of credentials due to inadequate implementation of identity and access controls is broken authentication and session management. An attack that takes advantage of a broken authentication is typically started by gaining access to login classes and credentials that are poorly managed in order to appear to be authenticated customers.

After manually spotting flaws in session management vulnerability user validation and verification, attackers use automated tools to retrieve additional data and take control of the application. Authentication and session management are essential parts of modern application security frameworks because attackers constantly look for ways to exploit security implementation flaws to gain access.

It can be challenging to scan for authentication and session management vulnerabilities in modern applications due to their complexity and integration. This post discusses vulnerabilities in session management and broken authentication, as well as methods and tools that are suggested for safely implementing them.

For the same user, a session is a series of events and transactions that occur

simultaneously. a one-of-a-kind Session ID (similar to cookies, URL parameters, authentication tokens, etc.) is provided to each user upon system login. During a valid session, communication between the user and the web app is made possible by this ID. Because many developers fail to develop the appropriate session parameters, it is simpler for a hacker to take over the session ID and gain unauthorized system access. Additionally, because some developers fail to set session time limits and rotation plans, attackers can impersonate users who are already logged in to the system.

2.1.3 Cross-Site Scripting (XSS)

Cross-Webpage Prearranging (XSS) attacks are a type of infusion in which harmful content is infused into benign and trusted websites. An attacker commits an XSS attack when they send malicious code to a different end client via a web application, typically as program side content. Anywhere a web application uses input from a client in the result it produces without approving or encoding it, flaws that allow these attacks to succeed are extremely inescapable.

A malicious message can be sent to a clueless customer by an aggressor. The program that serves the end customer has no real way to know that the content shouldn't be relied upon, so it will continue to execute the content. The vindictive content can gain access to any treats, meeting tokens, or other sensitive data held by the program and used with that website online because it believes the content came from a known source.

Attacks in which the injected script is permanently stored on the target servers, such as in a message forum, visitor log, comment field, or database, are referred to as "stored" attacks. The malicious script is downloaded from the server when the victim requests access to the stored data. Stored XSS is also known as Type-I XSS or Persistent XSS.

Blind cross-site scripting is one type of persistent XSS. It typically occurs when the backend application sends the attacker's payload back to the victim after saving it on the server. For example, feedback forms can be used by an attacker to send a malicious payload. The attacker's payload will be executed once the backend user or application administrator opens the submitted form via the backend application. For confirming blind cross-site scripting in real-world situations, XSS Hunter is one of the best tools.

These attacks are referred to as "reflected attacks" when the injected script is

reflected off the web server in an error message, search result, or other response that includes some or all of the request part sent to the server as input. The information is delivered to victims of reflected attacks in a different manner, such as via email or a different website. When a user is tricked into clicking on a malicious link, submitting a specially crafted form, or even just browsing to a malicious website, the injected code travels to the vulnerable website, reflecting the attack back to the user's browser. The browser then runs the code because it comes from a "trusted" server. Reflected XSS is also known as Type-II XSS or Non-Persistent XSS.

2.1.4 Insecure Direct Object Reference

Unreliable direct item references (IDOR) are a network protection issue that occurs when a web application designer utilizes an identifier for direct admittance toan internal execution object anyway gives no extra access control as well as approval checks. For instance, IDOR weakness would occur in the event that the URL of an exchange could be changed through client-side client contribution to show unapproved information of another exchange. Shaky direct item happens the designers use reference objects in URL. The aggressor can change the worth inreference protests and can see other data and afterward can do the catalog crossing attack.

2.1.5 Security Misconfiguration

Security controls known as security misconfigurations can be designed incorrectly or left unreliable, putting the designs and measurements in danger. Basically, a misconfiguration could result from setup changes that weren't properly reported, default settings, or a specific problem with any endpoint variable.

Misconfiguration weaknesses are problems with how things are set up that could be in parts or subsystems of programming. Web server software could send default client accounts that a cybercriminal could use to access the framework, or the product could have a predetermined arrangement of standard setup documents or catalogs that a cybercriminal could use.

Misconfigurations are what assailants use to influence their way to the intended targets, assuming that weaknesses are the entrance to the area. Because they can be found in any part of a company's frameworks, including its servers, working frameworks, applications, and programs, finding them is like finding a needle in a haystack. Associations fall victim to misconfiguration attacks due to a lack of

deceivability and integrated means of resolving configuration issues.

By employing standard change, contemporary local area foundations are particularly muddled and depicted; New business devices that are able to keep up with default configurations are one example of fundamental security settings that organizations can disregard without issue. Whether or not you deploy secure designs to endpoints, keep an eye on security controls and arrangements to see the inevitable setup stream. Frameworks change hands, and an entirely new framework is introduced into the organization; Misconfigurations are exacerbated every time patches are applied.

2.1.6 Sensitive Data Exposure

Any data that is intended to be protected from unapproved access is considered delicate information. Data that is actually recognizable (PII, such as Federal retirement aide numbers, banking data, or login credentials) can be considered delicate information. Clients are at risk for sensitive information exposure in the event that this information is accessed by an assailant as a result of an information breach.

Information is at risk of being exposed whenever an organization requires security measures. Development and security teams must first have a solid understanding of the ways in which that information is susceptible to openness in order to improve methods of relief against potential application attacks:

- **Information on the way:** When it reaches the application programming point of interaction (Programming interface), which enables applications to communicate with one another, information on the way is particularly vulnerable to attack. A man-in-the-center attack, which captures traffic and screens correspondences, is one type of attack that targets information along the way.
- **Information very still:** is stored in a framework, whether a computer or an organization. It is thought to be less powerless but simultaneously more significant without the threat of assaults passing by. In order to get close to home information, attackers use a variety of vectors, most frequently malware like PC worms or diversions. Either by clicking on malicious links sent via email or text message, or by directly downloading information from a malicious USB drive, these individuals gain access to frameworks that store information. In the event that information is stored on a server, attackers may be able to gain access to data stored in records in ways that go beyond the typical areas of access that are verified.

2.1.7 Missing Function Level Access Control

Clients can access resources that should be integrated or perform capacities that should be restricted thanks to the missing capability level gain passage to influence weakness. Although it is typically straightforward to protect assets and capabilities in the code or by design, it is not always straightforward to do so accurately. Attackers who suspect that assets or capabilities are not adequately protected should first get sufficiently close to the structure they want to attack. They should have permission to send genuine Programming interface calls to the endpoint in order to take advantage of this flaw.

An illustration of this flaw in an enrollment interaction designed to allow new customers to join a website is provided by OWASP. It would probably make use of a GET call from the programming interface, like this:

GET/programming interface/welcomes/{invite_guid}

The noxious client would get back a JSON with insights regarding the welcome, including the client's job and email. They could then change GET to POST and furthermore hoist their welcome from a client to an administrator utilizing the accompanying Programming interface call:

POST/programming interface/welcomes/new

{"email":"shadyguy@targetedsystem.com","role":"admin"}

Just administrators ought to have the option to send POST orders, however on the off chance that they are not as expected got, the Programming interface will acknowledge them as authentic and execute anything the aggressor needs.

2.1.8 Cross-Site Request Forgery

An attack known as Cross-Site Request Forgery (CSRF) allows an end user to carry out malicious activities on a web application that is currently being verified [9]. An attacker can trick users of a web application into performing actions of their choosing with just a little help from social design, such as emailing a connection or talking. If the victim is a regular customer, a successful CSRF attack can force them to perform state-changing actions like moving assets or changing their email address.

CSRF attacks target usefulness that alters the server's state, such as changing the victim's email address or secret phrase or making a purchase. Since the attacker does not receive the response, driving the victim to retrieve data does not result in the arrest of an attacker. As a result, CSRF attacks target demands that change with the

state.

An end user can be tricked into stacking data from or submitting data to a web utility using a variety of techniques. To launch an attack, the first step is to learn how to create a legitimate malicious request for our victim to carry out. Allow this to consider the model that goes along with it: Alice wants to transfer $100 to Sway using the bank.com web application, which is invulnerable to CSRF. All things considered, Maria is an assailant who wants Alice to send the money to Maria by tricking her. The advances that come after the attack will also be included.

The cash move activity could be reduced to a solicitation like: in the event that the application was intended to primarily utilize GET solicitations to move boundaries and carry out activities.

GET http://bank.com/transfer.do?acct=BOB&amount=100 HTTP/1.1

Maria has decided to use Alice to take advantage of this web application flaw because Alice is the person in question. Maria creates the accompanying maximum URL first, which will transfer $100,000 from Alice's record to Maria's. Maria takes the URL for the first order and uses herself as the recipient's name, increasing the total amount of the exchange at the same time:

http://bank.com/transfer.do?acct=MARIA&amount=100000

When Alice is signed into the bank application, the social designing component deceives Alice into stacking this URL. Typically, one of the accompanying procedures is used to conclude this:

- sending a spontaneous email with HTML content
- establishing an adventure URL or content on pages that are most likelyto be visited through the casualty while they're moreover doing internetbanking.

The endeavor URL can be veiled as a normal connection, empowering thecasualty to click it:

<ahref=http://bank.com/transfer.do?acct=MARIA&amount=100000">VIEW My Photos!

Or on the other hand as a 0x0 phony picture:

<img src="http://bank.com/transfer.do?acct=MARIA & amount= 100000" width="0" height="0" border="0">

Assuming this picture tag was remembered for the email, Alice wouldn't see anything. In any case, the program will in any case present the solicitation to bank.com

with practically no visual sign that the exchange has occurred.

The bank present utilizes post and the weak solicitation seems this way:

POST http://bank.com/transfer.do HTTP/1.1 acct=BOB & amount=100

Such a solicitation can't be conveyed utilizing standard an or IMG labels yetcan be conveyed utilizing a Structure labels:

<form action="http://bank.com/transfer.do method=POST">

<input type="hidden" name=acct" value=MARIA"/>

<input type="hidden" name="amount" value=100000"/>

<input type="submit" value="View My Photos"/>

</form>

This structure will require the client to tap on the submit button, yet this can beadditionally executed naturally utilizing JavaScript:

<body onload="document.forms [0].submit ()">

<structure…..>

2.1.9 Using Components with Know Vulnerabilities

Because it is so easy to exploit, this particular flaw poses a significant risk to the company. Since the attack methods are now available on the internet, the attacker simply needs to use them and can cause a negligible effect, serious or even total information split the difference, or lead to server/have takeover for associations. On the off chance that the attacker can figure out the weak parts that a specific application is using, it tends to be easily taken advantage of.

This flaw not only has the potential to bypass the application's security measures, but it also has the potential to serve as a catalyst for the development of additional attacks. For instance, programmers could direct a remote code execution or call a web administration without first obtaining authorization. Infusion, XSS, and broken admittance control are among the shortcomings caused by using weak components simultaneously.

Developers need to fully comprehend all of the dependencies they employ and think about the repercussions of doing so. In addition, all dependencies ought to be entered into an inventory system that can provide an easy overview of all the dependencies that are being used. Even though developers should be aware of which actions are carried out automatically, it is preferable to carry out all of these actions automatically.

These scans should be carried out on a regular basis, preferably automatically,

to avoid being overlooked. It is best to use an external system for these scans because of how the customer uses the web application or program. Additionally, this will guarantee that the resources required for these scans will not slow down any other servers.

2.1.10 Un-validated Redirects and Forward

Nullified Diverts and Forward Weakness, also referred to as URL Redirection Weakness on occasion, is a harmful program that can be found within the web application. The aggressor takes control of the URL in this vulnerability and sends it to the target. When the victim clicks on the URL, the site takes them to a harmful site or to a site that the aggressor believes the client should be taken to.

The attacker frequently takes advantage of this weakness by manually controlling the URL or by using a few tools like the Burp suite, which provides the attacker with a variety of methods from which he can control the URL to divert traffic.

2.2 Brute Force Attack

A brute force attack can be used to predict system login info, encryption keys, or discover invisible web page of the web applications. The attackers try to work over all possible password combinations expecting to guess correctly. Brute Force means that they use extreme forceful attempts to try and force their way into the target's private account(s). This is a popular, effective, old attack method. Cracking can take anywhere from a few seconds to many years based on the length and complexity of the password. [6] The consuming time (cracking time) is based on the number of characters and how numbers, letters and special characters are composed of are as shown in Figure 2.1.

Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	Instantly	Instantly	Instantly
5	Instantly	Instantly	Instantly	Instantly	Instantly
6	Instantly	Instantly	Instantly	1 sec	5 secs
7	Instantly	Instantly	25 secs	1 min	6 mins
8	Instantly	5 secs	22 mins	1 hour	8 hours
9	Instantly	2 mins	19 hours	3 days	3 weeks
10	Instantly	58 mins	1 month	7 months	5 years
11	2 secs	1 day	5 years	41 years	400 years
12	25 secs	3 weeks	300 years	2k years	34k years
13	4 mins	1 year	16k years	100k years	2m years
14	41 mins	51 years	800k years	9m years	200m years
15	6 hours	1k years	43m years	600m years	15 bn years
16	2 days	34k years	2bn years	37bn years	1tn years
17	4 weeks	800k years	100bn years	2tn years	93tn years
18	9 months	23m years	6tn years	100 tn years	7qd years

Figure 2.1 2020 passwords table used data from 2018 based on MD5 hashed passwords cracked by an RTX 2080 GPU (HIVE systems)

Brute force attackers have to take a bit effort to take advantages of these schemes.

How attackers benefit from brute force attacks are as following:

- Profiting from lean advertising or gathering activity data
- Robbery private personal data and valuables
- Transmission malware to cause disruptions
- Hijacking (take control of) your system for malicious activity
- Ruining a website's reputation

- **Profiting from lean advertising or gathering activity data:** Hackers can exploit a website alongside others to earn advertising commissions. Popular ways to do this include:
    - Putting spam ads (pop-up ads) on a well-traveled site to earn money based on the ad clicking time visitors' views.
    - Rerouting (url redirecting) a website's traffic to recognized ad sites.
    - Spreading a site or its visitors with activity-monitoring malware generally spyware. Data is sold to advertisers without your permission to help them develop their marketing.
- **Robbery private personal data and valuables:** Breaking into online accounts can be like cracking open a bank vault: everything from bank accounts to others information can be discover online. All of this is the criminal to steal identity,

money, or sell private credentials for profit. Sometimes, sensitive private databases from entire organizations can be discovered in highest point in an organization breaches.

- **Transmission malware to cause disruptions:** Attackers might redirect a website's traffic route to malicious sites to cause problem or practice their skills, Alternatively, they may directly affect a site with concealed malware (advanced persistent threat-APT) to be installed on web user's computers.
- **Hijacking (take control of) your system for malicious activity:** When one machine is not enough, attackers assign an army of unsuspecting devices called a botnet (short for robot network) to boost up their efforts. Malware can penetrate your computer, mobile device, or online accounts for spam phishing, increase brute force attacks and more. If there is not have an antivirus program, there may be more at risk of infection.
- **Ruining a website's reputation:** If you execute a website and become a target of smashing, a cybercriminal might determine to penetrate your site with obscene content (offensive description) including text, images, and audio of a brutal, pornographic, or racially offensive nature (raciest material).

2.3 Types of Brute Force Attacks

Different methods of brute force can be used to discover the sensitive data by using the following brute force popular methods:

- Simple (traditional) Brute Force Attacks
- Dictionary Based Attacks
- Hybrid Brute Force Attacks
- Reverse Brute Force Attacks
- Credential Stuffing

- **Simple (traditional) brute force attacks**: attackers attempt to logically guess the credentials without the need for help from software tools or other means. These can expose intensively simple passwords and PINs. For example, *"user12345"*as a simple password.

- **Dictionary based attacks:** a hacker aims a target and execute possible passwords in contrast to that username are known as dictionary based attacks. In brute force attacks, dictionary attacks are the nearly common basic attack tool. Dictionary are

used when the brute force attacks cannot afford to crack passwords in simple way. Most of the attackers execute over whole dictionary word lists and increase them with numbers and special characters or use unique words dictionaries, but this attack is complicated.

- **Hybrid brute force attacks:** Mixing dictionary and brute force attacks form a hybrid attack usually. Passwords that merge with random common words can be found out with these attacks. For examples, passwords such as *NY1993* or *US1234*.
- **Reverse brute force attacks:** These attacks start with a known password by altering the brute force attack method. Then attackers explore million usernames till they get a match. Reverse brute force attacks can be executed with the stolen passcodes (passwords) that obtained from online existing data cracks.
- **Credential stuffing:** If the attacker has a pair of username and password which can be accessed for one of the websites, they will try to use that combo in lots of any others sites. Users have been known to repeat login info through many websites, they are the private mark of an attack.

2.4 Tools Aid Brute Force Attempts

Guessing a password by using tools can be faster than guessing for a particular user or website.

Automated tools help use rapid-fire guessing that is organize to build every possible password and try to use them. A dictionary word password can be found within one second with brute force hacking software.

The following tools can have cope programmed in them to:

- Thwart many computer protocols (like MySQL, FTP, Telnet and SMTP)
- Allow hackers to crack through wifi.
- Discover weak passwords
- Password encryption and decryption
- Convert words into "Thanksforwatch" becomes "Th@nk4watch," for example.
- Execute all possible sequence of characters.
- Generate dictionary-based attacks.

Some software tools explore pre-computed compilation of plain text and matching cipher text (rainbow tables) for the inputs and outputs of common hash functions. To convert passwords into long, fixed series length of letters and numerals,

the algorithm-base encryption called “hash functions” are used. In other words, rainbow tables can speed up brute force attack.

2.4.1 GPU Speeds Brute Force Attempts

To execute brute force password software, tons of computer brain power is needed. Linking together with the CPU and graphics processing unit (GPU) raise the computer performance. The system can handle complex tasks at once by adding the thousands of multi-cores in the GPU for processing. Analytical process, many types of engineering, and other compute-intensive applications can be used GPU processing. GPU processing can make the cracking password 250 times quicker than a CPU alone. [11]

2 billion possible password combinations can be produced with a six-character password. It only takes more than two years to tries 30 passwords per second by using powerful CPU. But by adding a powerful single graphics card (GPU) with the same level of computer test 7,100 passwords per second can produce and takes 3.5 days to crack.

2.5 Steps to Protect Passwords for Professionals

To put ourselves and our network secure, we will take our safeguards and support others to do so. Network security systems and user actions systems will both need guidance.

A few overall pieces of advice for IT experts and users are:

- Use strength username and password. Defend with credentials that are more secure than *defaultuser* and *password111* to protect the attackers. The harder for anyone to intrude the password if the stronger this combination is.
- Delete any inactive old accounts with powerful permissions. These are the cyber doors with weak locks that make cracking in easy.

Passive Backend Attack Protections for Passwords

- **Strong encryption rates:** System administrators should use the encryption algorithm with 256-bit encryption with the highest encryption rate possible difficult for brute force to operate. The larger bits in the encryption, the harder for the password to crack.

- **Add salt to the hash:** Randomize password hashes by adding an arbitrary string of letters and numbers (called salt) to the password itself. This random string should be keep in different database, retrieve and added to the password before it is hashed. With salting the hash, the same password has different hashes.
- **Two-factor authentication (2FA):** Two-step authentication and IDS should install to detect brute force attacks by the administrators. Users have to add a login attempt with a second factor, such as a USB security key(also called U2F key) or biometrics (physical characteristics) scan.
- **Limit login attempts reloaded:** Brute force attacks vulnerability can be reduced by limiting the number of logins attempts. For example, only three logins attempts are allowed to enter the correct one before locking out the user for few minutes can cause pretty long delay and let attackers to run to easier target sites.
- **Account lockout after unsuccessful login attempts:** if an attacker can repeatedly keep retrying passwords after a temporary interlock lockout time, they can return to try again. Locking out the account and informing the account user to notify IT technician for an unlock will stop this activity. Relatively short lockout timers are more accessible for users, but comfort can be a susceptibility. To solve this, consider using the long-term lockout if there are max- failed logins scenario after the short login attempts.
- **Login attempt throttle rate:** By putting some space between each single login, it can further delay an attacker's works. Once a fail login attempt, a timer can restrict access until a short space of time has run. This will delay for real-time monitoring team to mark and work on discontinuance this attack. Some hackers might not keep trying if the wait is needed more time.
- **Enable Captcha after several login failures:** hand-operated identification stops robots from brute-forcing into the data. Many types of Captcha are, typing the text in an image again, checking a tick box, or describing objects in photos. Use block IP address list to block already known attackers. This block IP address list need to be updated regularly.

Active IT Support Protections for Passwords

Password management in education: user action is necessary to security of password. Initiating on safe trials and tools to assist them hold the passwords. Kaspersky Password Manager services allow users to keep their sophisticated, long passwords in an vault of encrypted instead of insecurely writing them down. To help them secure about their passwords, benefit tools should be put to theirs hands.

Real time monitoring accounts for unusual activity: Weird login locations, too many failed login attempts etc. Need to find suspicious actions and need to take efforts to block them in real-time. Checked to see if an IP address is blacklist, account lockout, and inform users to determine that account is authorized.

2.6 How Users Can Strengthen Passwords Against Brute Force Attacks

Users can do a lot of prevention in the IT world. The best guard against password attacks is that how passwords are as strong as they can be.

Brute force based on cracking time of the password. So, the main is to make the password delays these attacks as much as possible, because if it takes too much time to crack most hackers will turn around.

Ways which can strong passwords against brute attacks are:

- **Longer passwords with different character types.** Users should select 10-character passwords that include symbols or number and special characters. By doing so it generates 171.3 quintillion (1.71×10^{20}) possible passwords. Using a GPU processor that can perform 10.3 billion hashes per second, cracking the password would take nearly 526 years although a supercomputer could crack in a few weeks. As this logic, including more characters makes the password stronger.
- **Complicated passwords.** All the websites cannot accept long passwords, which means that choose complex passphrases rather than single words. Dictionary -based attacks make a breach nearly effortless because they are built with single word phrases. Passphrases — passwords contained large number

of words or segments — should be sprinkled with extra characters and special character types.

- **Create passwords construction rules.** Using trimmed words like using: newyork" with "ny" can be strong passwords but anyone else can read them. For example, using only the first two or three letters of each words or avoid using vowels.
- **Avoid using commonly used passwords.** The mostly used password should not be created and it should be change quite often.
- **Use different passwords for every site.** Never reuse a password of one site to other sites as well. Using unique username and passwords can enhance the security.
- **Use a password manager tools.** Password manager can create and monitors the online login info. All the password that are used in many accounts can be accessed via logging into the password manager. In password manager, only a long, sophisticated passphrases for all sites are needed and only to memorize one password of password manager.

2.7 Dictionary Based Attack

A dictionary attack uses every word in a dictionary as a password to break into a password-protected computer system, web application, computer network or other IT resources. A dictionary attack can also be used in an attempt to find the key necessary to decrypt an encrypted message or document.

Dictionary attacks work because many computer users and businesses insist on using ordinary words as passwords. These attacks are usually unsuccessful against systems using multiple-word passwords and are also often unsuccessful against passwords made up of uppercase and lowercase letters and numbers in random combinations.

In systems with strong password requirements, the brute-force method of attack, in which every possible combination of characters and spaces is tested up to a certain maximum length, can sometimes be effective. However, a brute-force attack can take a long time to produce results.

In the predetermined password library, strong, randomized password cannot be easily guessed because they are not included in the pre-created password library.

Because a dictionary attack's guess attempts are limited to a preselected list, it is essentially impossible to crack non-predictable passwords.

2.8 Work of Dictionary Based Attack

A dictionary attack requires a predefined lists of word list library and phrases to predict possible passwords that are varied by the region. It generates basically under the user's common lists of password such as "password," "123abc" and "1234."

Attackers created word list dictionaries related to country, languages spoken, organization name, cities, addresses and other regionally specific items. They are quite large, although these lists are not as wide as those of other brute-force attacks. By practical means, it cannot be processed and tested all these passwords manually. Attackers use password libraries or other brute force attack tools as supporting programs because the extra technology is needed to boost the process.

Dictionary attacks are classified based on the online and offline state of the target user account, computer system network or device. In online dictionary attack, the attack must be aware of the number of login tries when they try to guess the correct password. When website administrator, user account administrator, user or intrusion detection system may detect the number of tries, block the attack from the system. If not the attacker can ne locked out of the system.

Dictionary attacks can easily guess with a short list of mostly used passwords. Smart attackers can also be able to modify the detection settings or login attempt limits.

For offline attacks, there has few limits to try. Offline dictionary attack can be only need to access the password storage file from the target system because it is offline attacks.

2.9 Brute-force attack vs. dictionary attack

The number of password permutations [3] that are attempts is the main distinct between a brute-force and a dictionary -based.

Brute-force attacks

A brute-force attack uses a systematic approach to generate all possible passwords but it needs a lot of time to fill in.

The example of difference is a five-digit combination lock. An attacker will attempt all possible variation for the lock each value from zero to nine has 100,000 possible changes exactly. [7]

The brute force attack is shown in Figure 2.2.

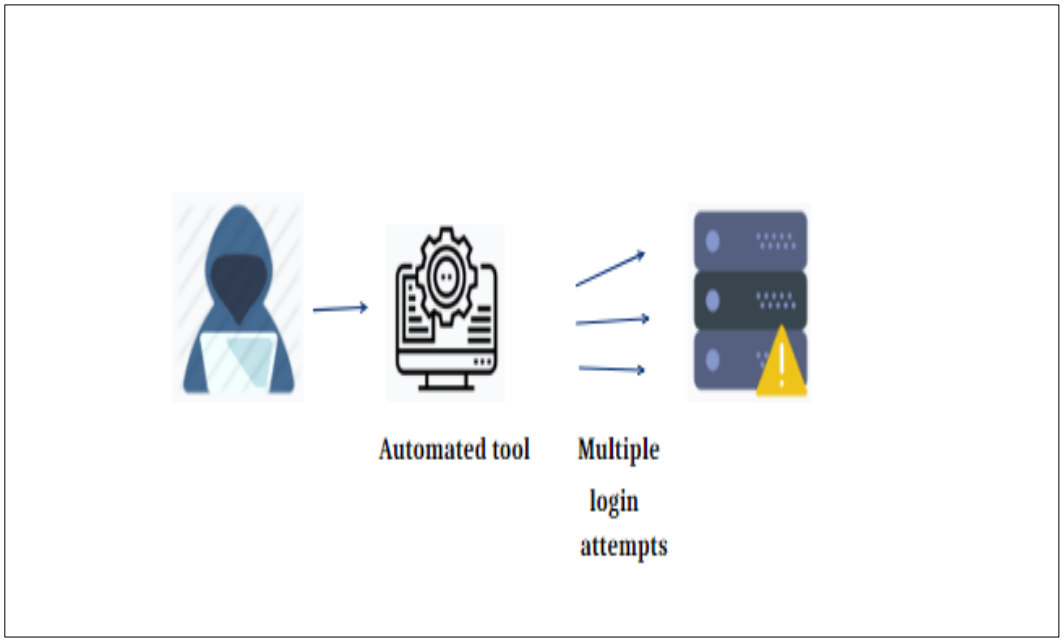


Figure 2.2 Brute Force Attack

Dictionary attacks

A dictionary attack is using a list of likely passwords and attempting to break into system. These attacks are more focused than brute-force attacks. Instead of trying to input every permutation, an attacker who is using a dictionary approach will attempt will attempt all the permutations in its predefined library.

Sequential key like “12345” and static key like “00000” will be tested. The dictionary attack cannot predict this if the five-numerals permutation is significant. In the same way as phishing attacks, the dictionary attack is assumed that a number of the users or accounts which use five-digits code is vulnerable and easy to attack.

How dictionary attack works to guess the password is shown in Figure 2.3.

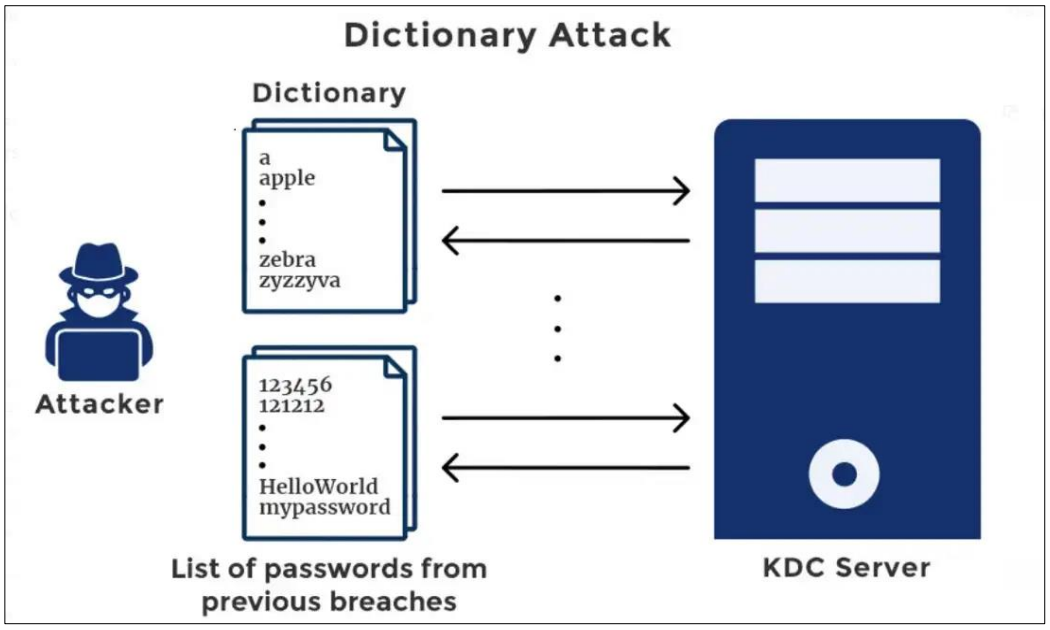


Figure 2.3 Dictionary Attack

2.10 How to protect yourself against a dictionary attack

Choosing passwords or keys wisely and limiting the attempts allowed in a given amount of time can reduce to nearly zero is vulnerable to password or decryption key assaults. For a system immunity to dictionary attacks and to brute-force attacks need the following three stages:

1. Three password logins are allowed;
2. Limit time to 15 minutes the logins are allowed; and
3. The key should be a strong, meaningless mix of letters, numbers and special characters.

The dictionary attack is often used by the email spammers as a message of email addresses that contain names, it sent the @ email symbol and the domain name. The given names such as Susan, John, Karina or Rena, or each letter is paired by Semitic; combining a domain name with surnames such as karrie, kate or smith, are usually successful.

CHAPTER 3
DESIGN OF THE PROPOSED SYSTEM

Nowadays, many people do a lot of things on the internet. Among these purposes, the most used internet technology is the web application. A web application is combined with a web server (server-side) and web browser (client). When people access a web application from any one browser, firstly they send a request to the web server and then this web server responds this request to the web application server and processing continued tasks Today, web applications are popular for people because these have many advantages: easily use and cost effective for users. Maintaining web application security is the important case for users because web application may have vulnerabilities. Web application vulnerabilities are weakness of this web application and can find many kinds of reasons. For example, application developer errors within coding, application design weakness and so on. So, attackers can be tried by using these vulnerabilities to exploit system for getting privileges and personal information. This system will study the password attacking using brute force attack and dictionary based.

A brute force tries to guess the private user data, login information and encryption keys as a trial approach. The attacker tries to enter mixture of usernames and passwords until they find the correct one. If there have been passwords, brute force attacks have been around. A few numbers of password are guessed by the brute force on every second. Easy passwords, such as upper- and lowercase letters and those using common numbers like ‘123456789’ or ‘passwords,’ can be exploited in minutes.

A dictionary attack used dictionary words as a password and systematically entering every word to a computer system, computer network and other IT resources. Although dictionary attacks used numbers and words of dictionary, but today it also uses passwords library that are stolen earlier.

3.1 Combination Algorithm for Brute Force Attack

```
function combination:
pass in: inputArray, combinationArray, start, end, index, r
    if index is equal to r:
        for each element in combinationArray:
        print each element
        return
    for i = start:
        if i <=end and end – i+1 > r - index:
        combinationArray[index] = inputArray [i]
    call combination function again with updated parameter
```

The Combination is a kind of arrangement of some given objects. [2] In mathematical terms, Combination is a set of choices/selection of items from a unique set of items/objects. For example, you are given a bag with 5 different colors and asked to generate a pattern with any 3 colors. You can also pick any 3 colors out of 4, then arrange them in different orders.

Let's assume the colors are RGYBI (R= Red, G= Green, Y= Yellow, B= Blue, I= Indigo). So, the possible pattern can be RGB, RGY, etc.

Formula of combination:

$$C = \binom{n}{r} = \frac{n!}{r!(n-r)!} \qquad \textbf{Equation 3.1}$$

- This has 5 colors meaning n = 5, and at any given time, we need to pick any 3. So, r = 3.

- After calculating, we get,

$$_5C_3=\binom{5}{3}=\frac{5!}{3!(5-3)!}=\frac{120}{6*2}=10$$

- total of **10** color combinations

Example: color combination

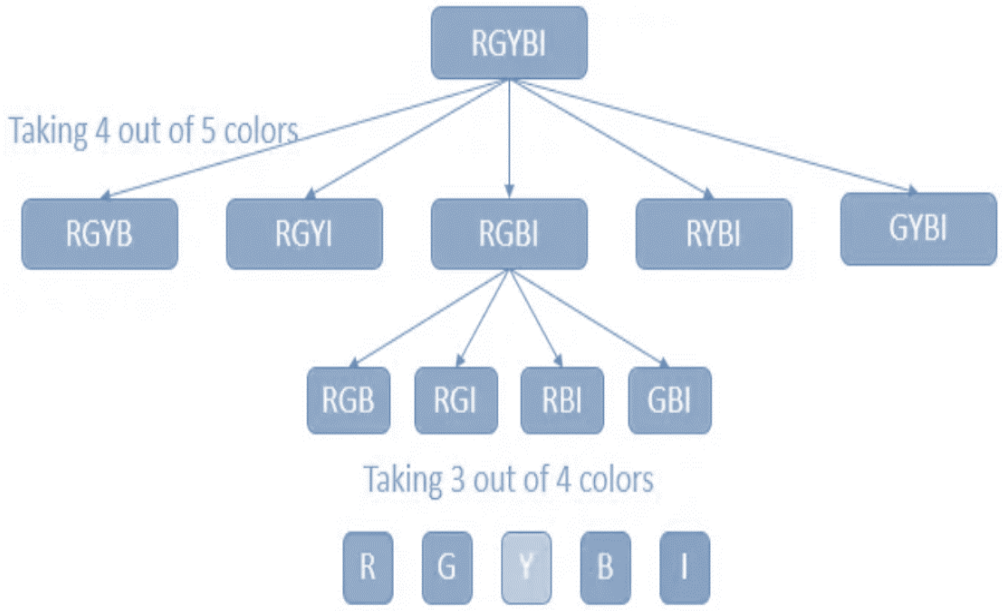


Figure 3.1: Sample Password Combination of Brute Force Attack

- Take any 4 colors out of 5 and list them [1]
- From each block of 4 colors, pick any 3 and list them all.
- For example, we only picked “RGBI” in the figure and showed 4 combinations.

Combinations:

- RGY
- RGB
- RGI
- RYB
- RYI
- RBI
- GYB
- GYI
- GBI
- YBI

3.2 Approximate Time, Combination and Password Length for Brute Force

When creating combinations of password, the amount of password combinations, password length and approximate time may be depended on the memory and speed of the computer. [5]

The following are the types of character for combination brute force.

- Numbers (from 0 to 9)
- Letters (from A-Z and a-z)
- 32 special characters

The formula for possible combination with for all the chosen characters for brute force is:

$$\textbf{Possible combinations} = \textbf{possible number of characters Password}^{\textbf{Length}}$$

Equation 3.2

Some of the possible combinations for above formula shown in the following table.

Table 3.1 Approximate time for brute force based on the character length

Password contains	Number of combinations	cracking time based on the generation of 2 billion key per seconds
5 characters (3 lowercase letters, 2 numbers)	$36^5 = 60,466,176$	60,466,176 / 2,000,000,000 = 0.03 seconds
7 characters (1 capital letter, 6 lowercase letters)	$52^7 = 1,028,071,702,528$	1,028,071,702,528 / 2,000,000,000 = 514 Seconds = approx. 9 minutes

8 characters (4 lowercase letters, 2 special characters, 2 numbers)	68^8= 457,163,239,653,376	457,163,239,653,376 / 2,000,000,000 = 228,581 Seconds = approx. 2.6 days
9 characters (2 uppercase letters, 3 lowercase letters, 2 numbers, 2 special characters)	94^9= 572,994,802,228,616,704	572,994,802,228,616,704 / 2,000,000,000 = 286,497,401 Seconds = approx. 9.1 years
12 characters (3 uppercase letters, 4 lowercase letters, 3 special characters, 2 numbers)	94^{12}= 475,920,314,814,253,376,475,136	475,920,314,814,253,376,475,136 / 2,000,000,000 = 237,960,157,407,127 Seconds = approx. 7.5 million years

3.3 Dictionary for Attack

The dictionary for the proposed attacking system is downloaded from "CrackStation's Password Cracking Dictionary Site".[4] Sample contents of downloaded dictionary is shown in following Figure 3.1.

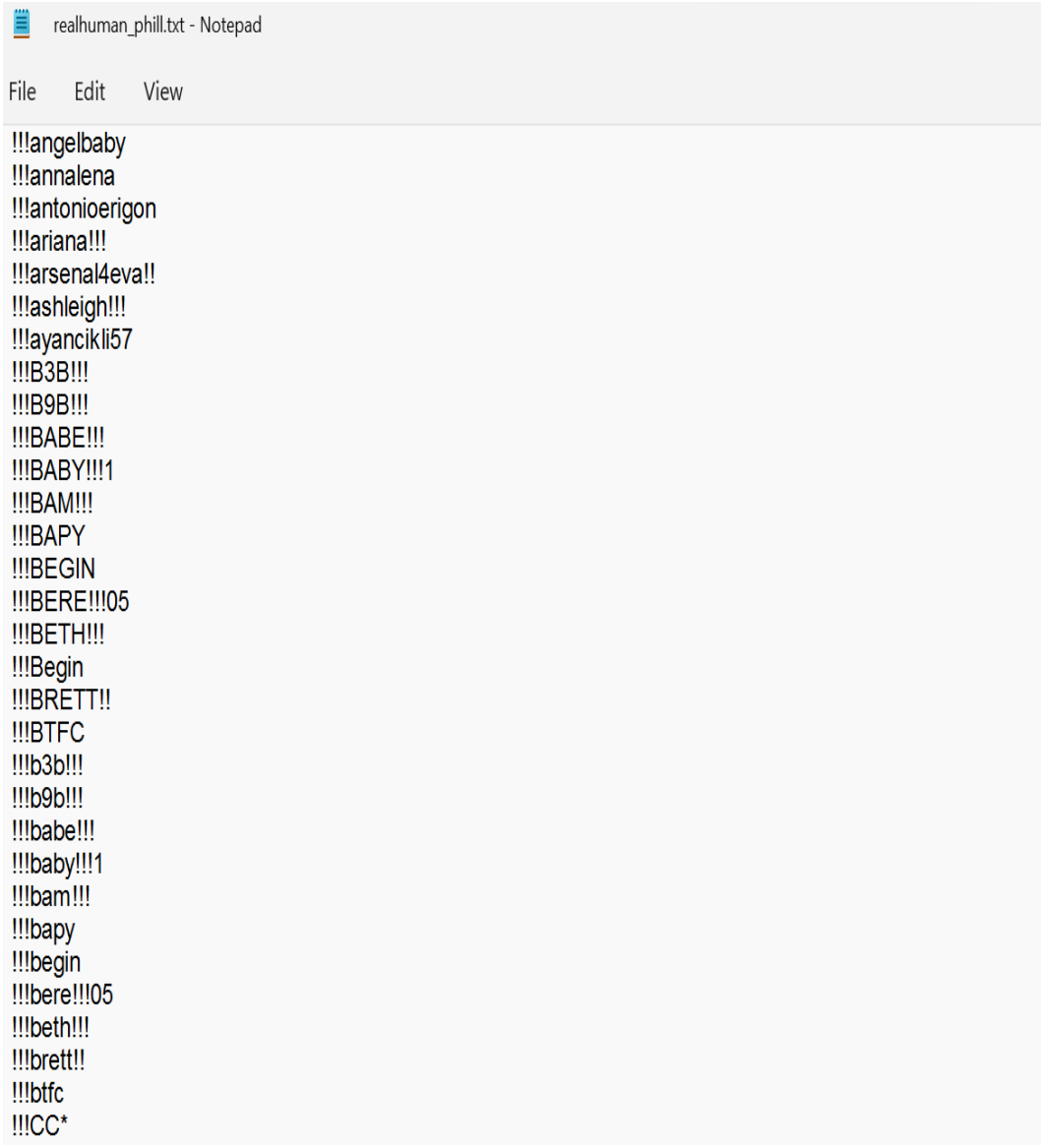


Figure 3.2 Sample Password Contents of Dictionary Based Attack

3.4 The System Flow

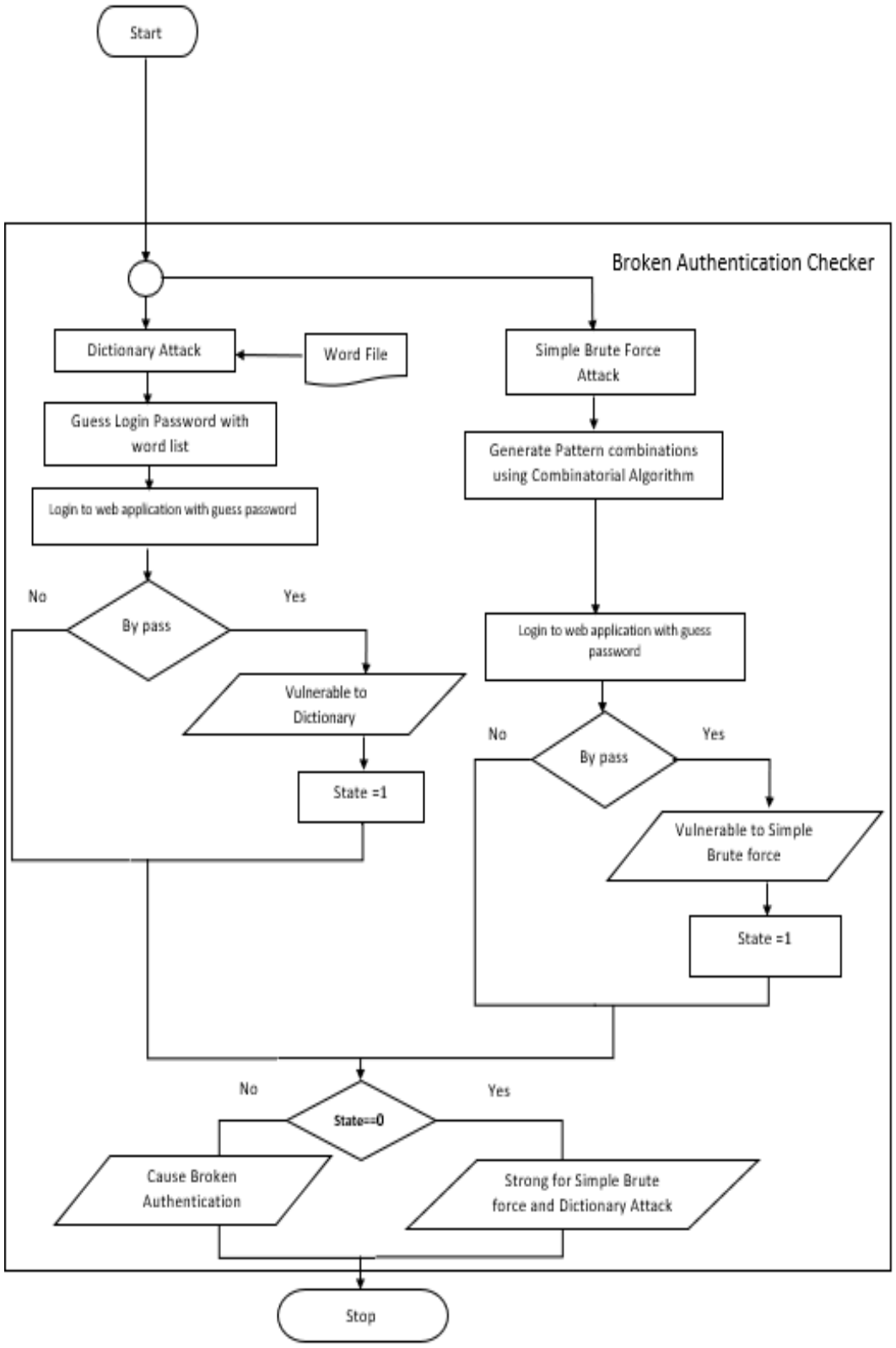


Figure 3.3: The System Flow

In this Figure 3.3, the system flow shows that the password guessing process will guess the vulnerable website's password by two ways: dictionary based and brute force attack. Set the state to 0 and start checking with dictionary and brute force attack. In dictionary attack, the proposed attacking system will guess the password using the words in downloaded word file. Bypass that website with the guess password, it is set the state to 1 and if it is not passed, it is set the state to 0. And at the same time, generate the pattern combination using combination algorithm in brute force. Bypass that website with the guess password, it is set the state to 1. If it is not passed, it is set the state to 0. After bypassing with one of two methods, return the message "Strong for simple brute force and dictionary attack" depending on the state. If not return the message "Cause broken authentication".

3.5 The Evaluation of the Proposed System

This proposed system guesses the password of the vulnerable web site using Dictionary based attack and Brute force attack. The experimental result is based on length of password, time consuming and type of methods. The 6 characters' length of password and only numbers (for example - 456789), the possible combinations for brute force is 10^ 6 =1,000,000 (for 10 digits from 0 to 10).

CHAPTER 4
IMPLEMENTATION OF THE PROPOSED SYSTEM

This chapter contains the implementation of proposed system, how the system is designed and the testing process of the system and the evaluation results of the system as a table in milliseconds and how the target web application is designed and the screenshots of the target web application.

4.1. Experimental Setup

This chapter consists of System Implementation, System Design and Testing Results of the proposed system. The system may attack the pre-created web application which is located on the Microsoft SQL server with the simple brute force and dictionary attack as a white hacker.

4.2 Implementation of System Design

The main form of the proposed system was designed with three action buttons called Dictionary Based Attack and Brute Force Attack and Evaluation.

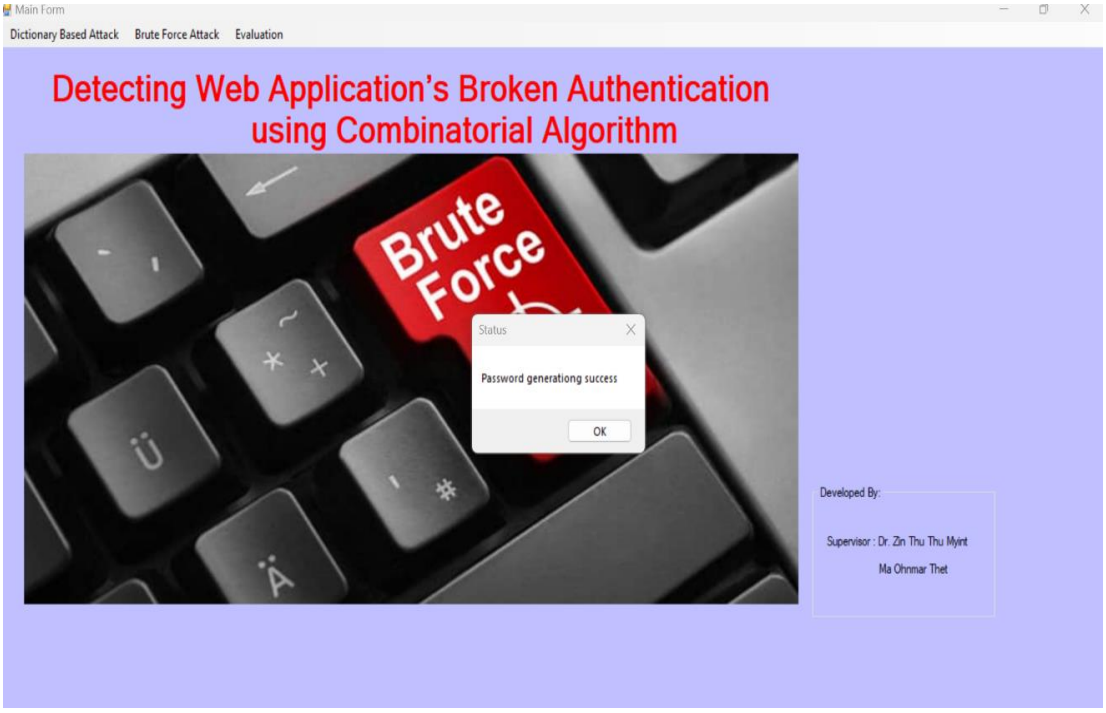


Figure 4.1 Main form of the proposed system

When the system is executed, the main form is displayed. Then start attacking with Dictionary Based Attack by clicking the Dictionary Base Attack option in the main

form. In this stage, the dictionary attack starts with the possible password from the downloaded password text file. When the password guessing is successful, it displays the message box “Attacking Success, Time of guess in **.....**milliseconds and also shows the success guess password**”** . Figure 4.2, 4.3 and 4.4 shows that how dictionary attacks are generated in the system.

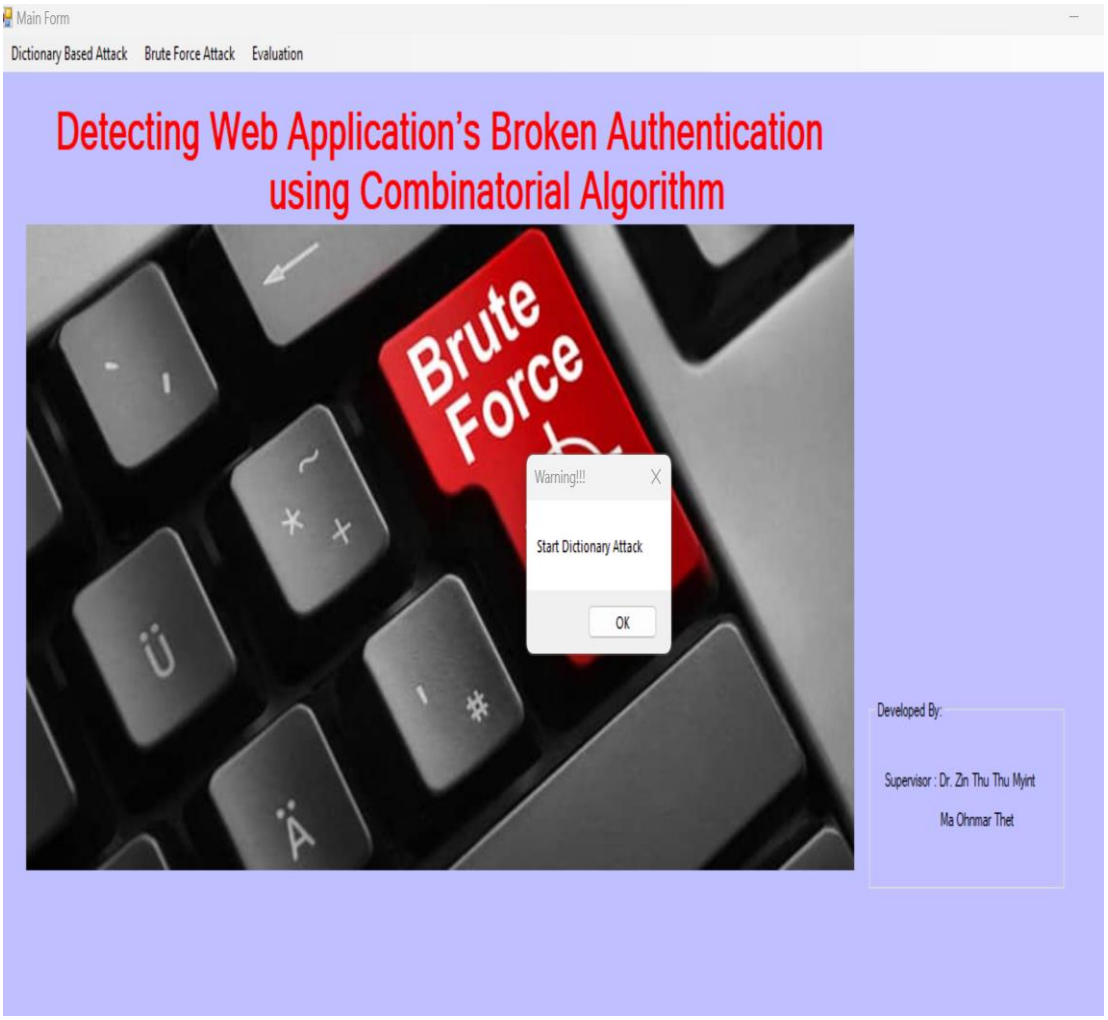


Figure 4.2 Status for Starting Dictionary Attack

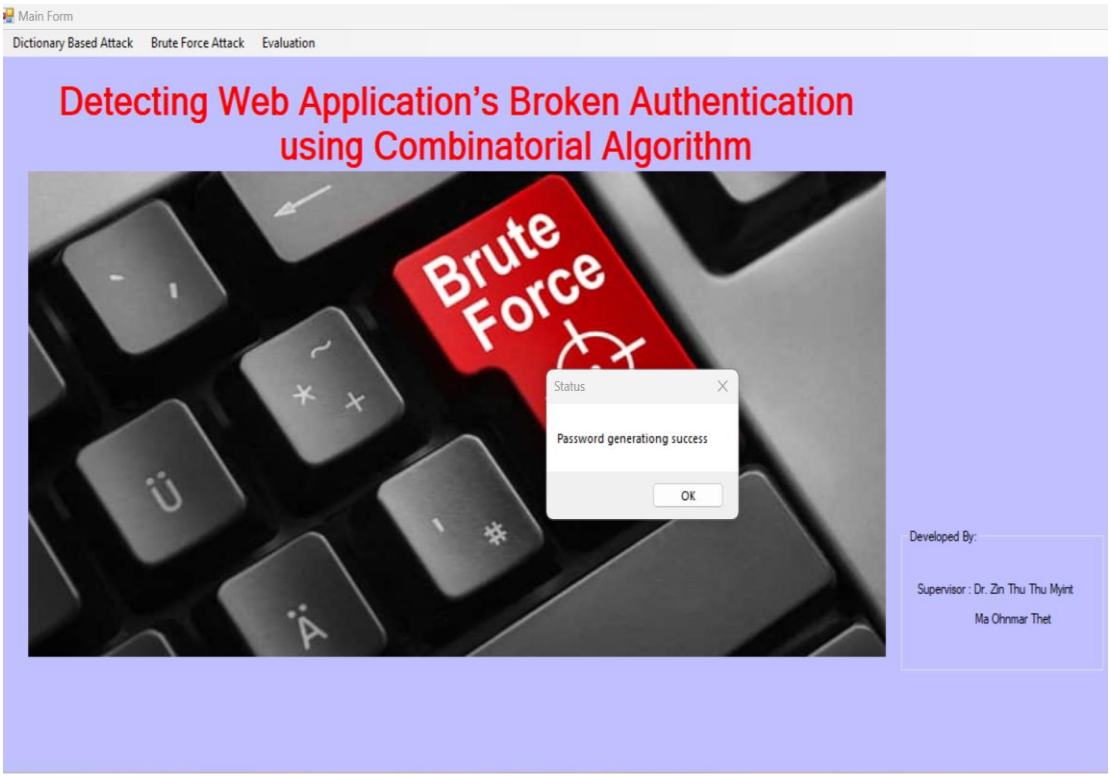


Figure 4.3 The Status for Dictionary password generating success

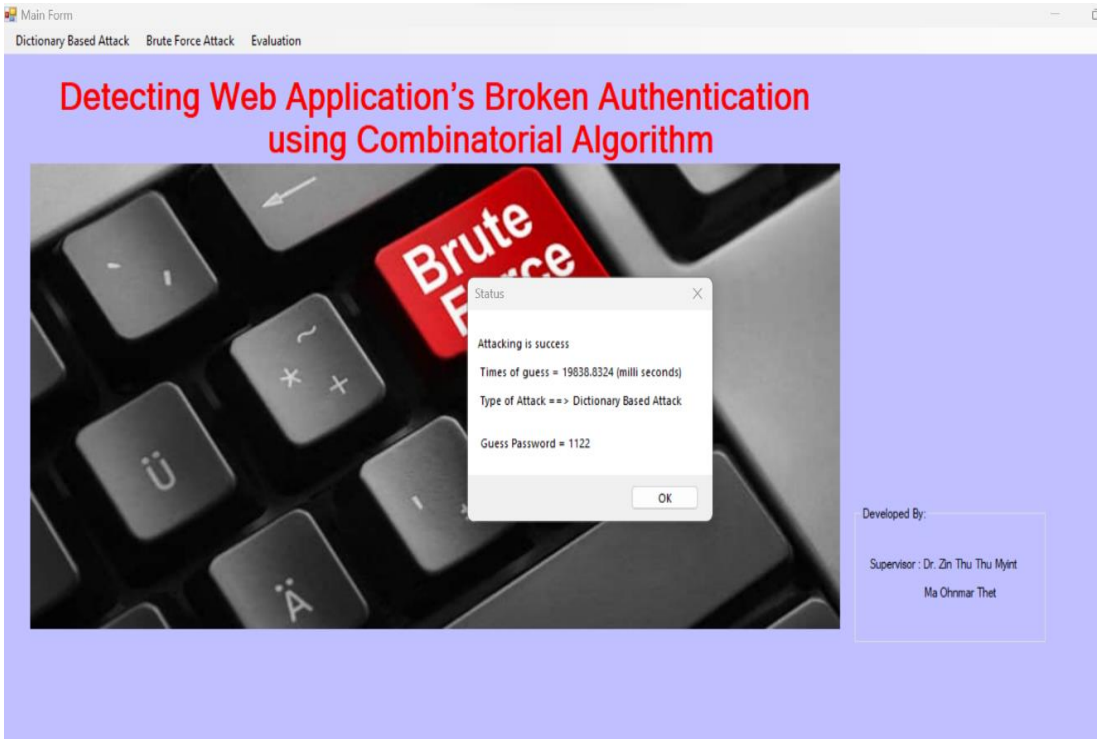


Figure 4.4 The status box for success dictionary attack with guess password

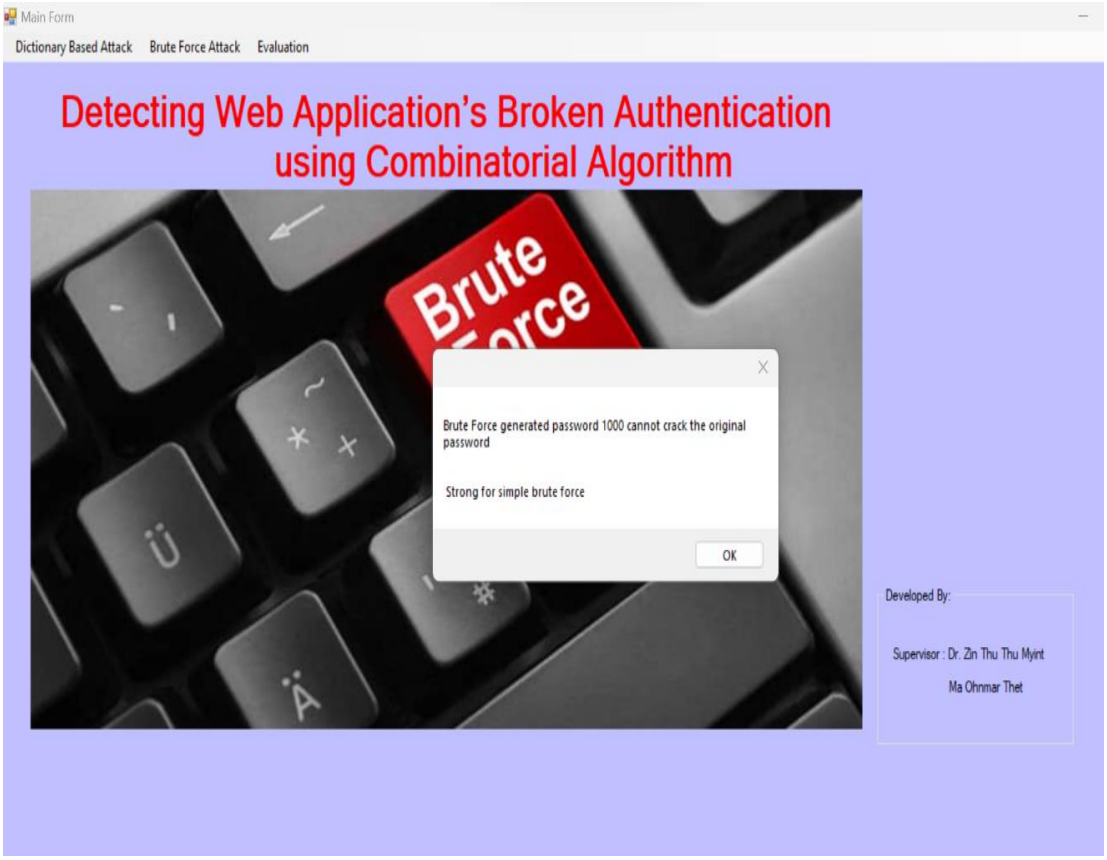


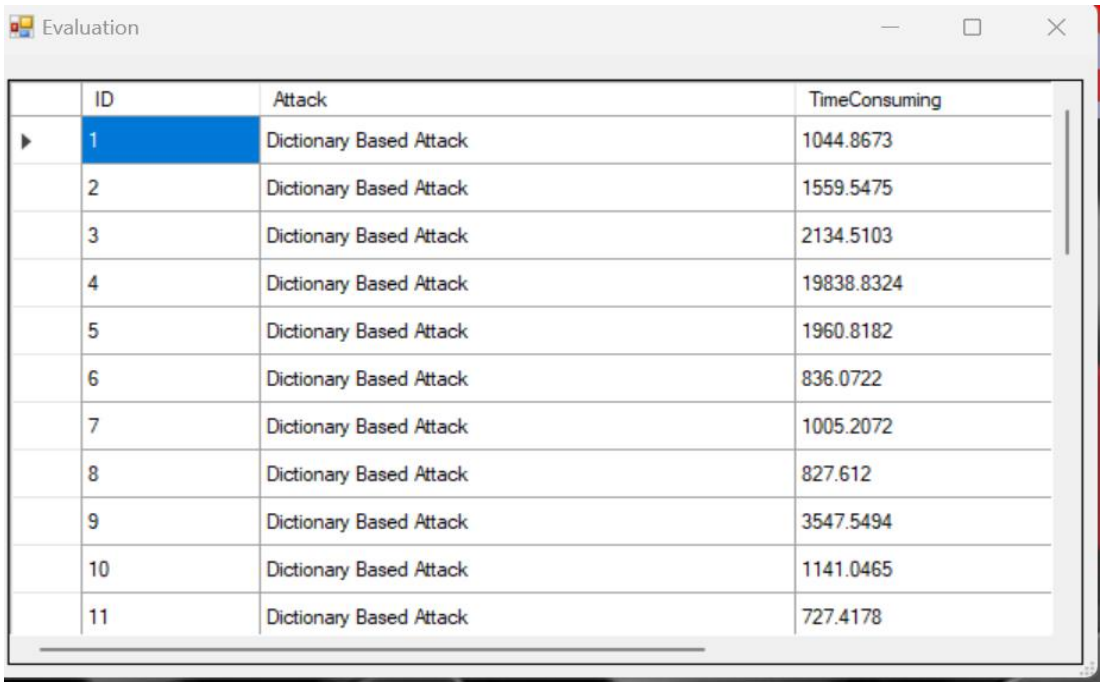
Figure 4.5 The status for Brute Force generating process

In Brute force generating process, as shown in Figure 4.5, the process may start by clicking the Brute Force Attack on the main form. Brute force attack makes the combination of characters and generating the guess password of the target web application based on the limited time. When the guess password is not generated, it showed the message of "Brute force generated password 1000 cannot crack the original password, Strong for Brute Force". Brute force attack can crack the almost the passwords but it depends on the time, memory and speed of the computer system. So, due to the consuming time, the brute force attack may work for 15 minutes and also make a combination of 1000 passwords.

When generating the guess password with both or one of two methods, the guess password will be manually entered to the target website login page and check the guess password is useful or not. When the guess password is access to the login page of the target web application, the system is compromised.

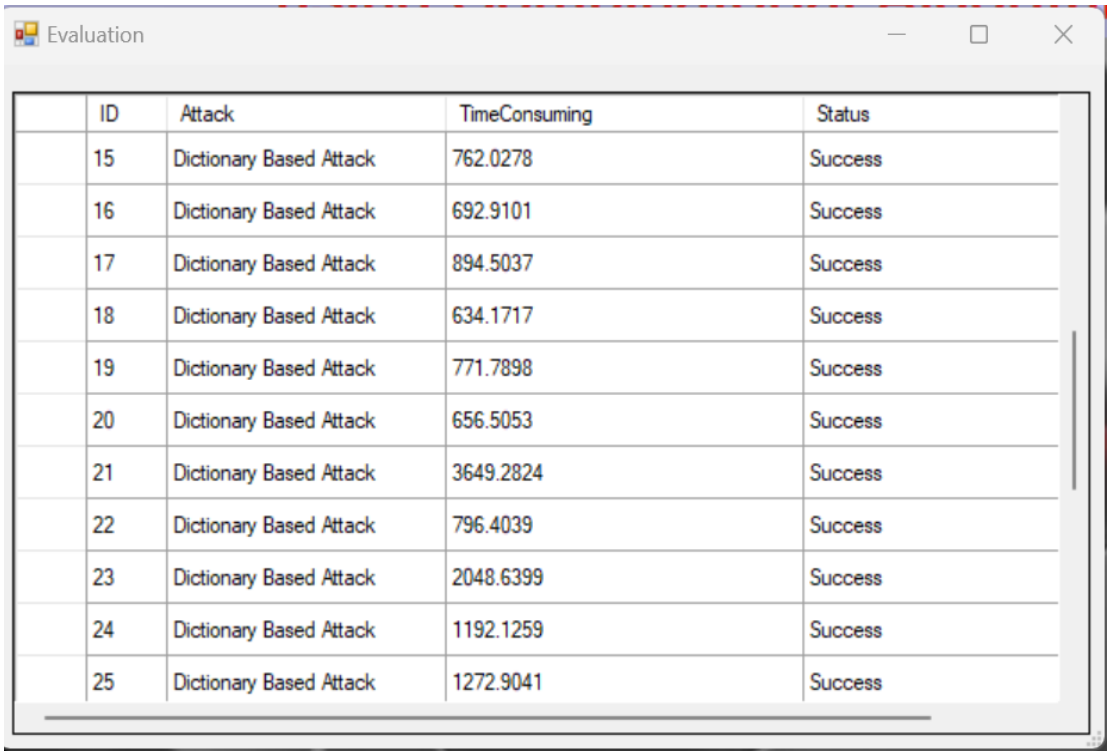
If none of the two attacks cannot access to the target web application, the web application can be assumed as strong enough for broken authentication.

After attacking with these two methods, the result of the evaluation can be checked by clicking the Evaluation option on the main form. The following three figures are the some of the evaluation of dictionary attacks based on the milliseconds.



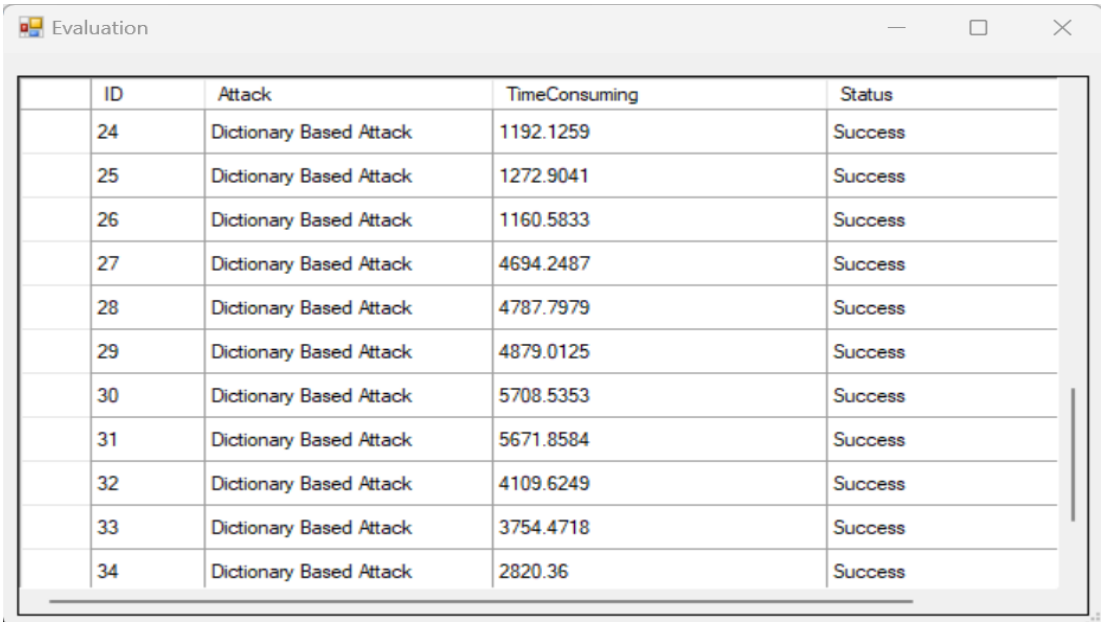
ID	Attack	TimeConsuming
1	Dictionary Based Attack	1044.8673
2	Dictionary Based Attack	1559.5475
3	Dictionary Based Attack	2134.5103
4	Dictionary Based Attack	19838.8324
5	Dictionary Based Attack	1960.8182
6	Dictionary Based Attack	836.0722
7	Dictionary Based Attack	1005.2072
8	Dictionary Based Attack	827.612
9	Dictionary Based Attack	3547.5494
10	Dictionary Based Attack	1141.0465
11	Dictionary Based Attack	727.4178

Figure 4.6 Evaluation Results for Dictionary Based Attacks (1)



ID	Attack	TimeConsuming	Status
15	Dictionary Based Attack	762.0278	Success
16	Dictionary Based Attack	692.9101	Success
17	Dictionary Based Attack	894.5037	Success
18	Dictionary Based Attack	634.1717	Success
19	Dictionary Based Attack	771.7898	Success
20	Dictionary Based Attack	656.5053	Success
21	Dictionary Based Attack	3649.2824	Success
22	Dictionary Based Attack	796.4039	Success
23	Dictionary Based Attack	2048.6399	Success
24	Dictionary Based Attack	1192.1259	Success
25	Dictionary Based Attack	1272.9041	Success

Figure 4.7 Evaluation Results for Dictionary Based Attacks (2)



ID	Attack	TimeConsuming	Status
24	Dictionary Based Attack	1192.1259	Success
25	Dictionary Based Attack	1272.9041	Success
26	Dictionary Based Attack	1160.5833	Success
27	Dictionary Based Attack	4694.2487	Success
28	Dictionary Based Attack	4787.7979	Success
29	Dictionary Based Attack	4879.0125	Success
30	Dictionary Based Attack	5708.5353	Success
31	Dictionary Based Attack	5671.8584	Success
32	Dictionary Based Attack	4109.6249	Success
33	Dictionary Based Attack	3754.4718	Success
34	Dictionary Based Attack	2820.36	Success

Figure 4.8 Evaluation Results for Dictionary Based Attacks (3)

The following figures are the login page of the target web application. The target web application is developed as an online book shopping website. This website includes the login page, Home page, Category, History, Search, About, User Registration and Logout page. There is also a submenu as category such as Children Book, Health and Fitness, Business, Fiction, Programming, Science Book and History Books. Only the authenticated user can access the web application by entering the correct password to the login page.

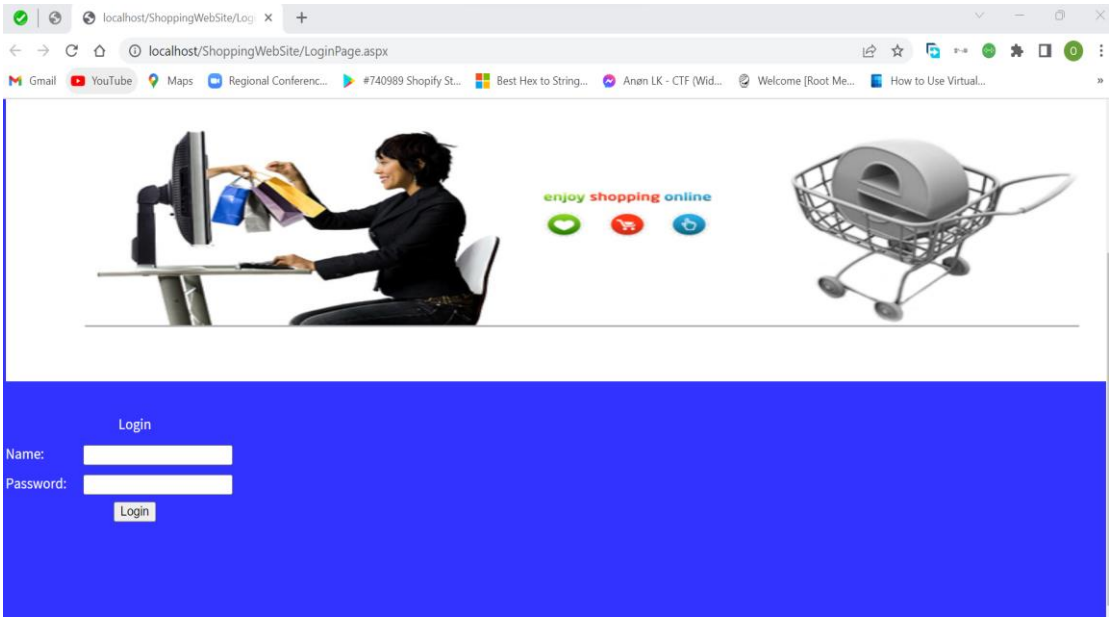


Figure 4.9 Login page of the target online book shopping web application

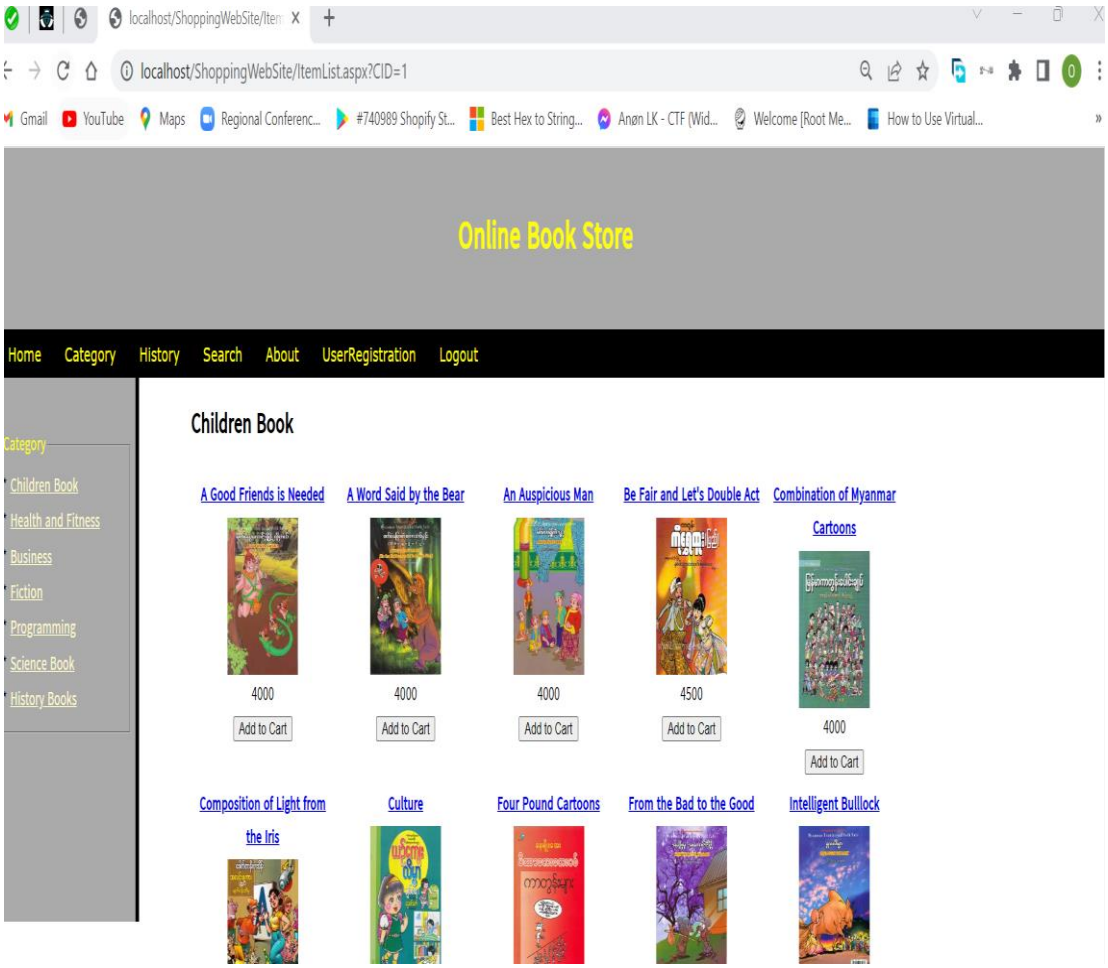


Figure 4.10 Main page of the online book shopping web application

4.3 Experiment Results

The results of the proposed system for time consuming for brute force and dictionary attacks are shown in the following Table 4.1. Because of the time and password length limitations of the brute force attack, the password length is tested from length 4 to 8 (including numbers, uppercase and lowercase letters and special characters). The estimated cracking time for Dictionary Based Attack and Brute Force Attack is calculated on milliseconds and the Brute Force Attack also includes the addition of combination time in the cracking time.

Table 4.1 Estimated Cracking Time based on the password length

Attacks / Password Length	Brute Force Attack(ms) (combination time+ estimate cracking time)	Dictionary Based Attack(ms)
Password Length 4	8784.378529	1944.3195
Password Length 5	85212.78191	924.1135
Password Length 6	3449000	1729.2016
Password Length 7	32,423,879.70963	805.0287
Password Length 8	3,047,844.69	2223.8646

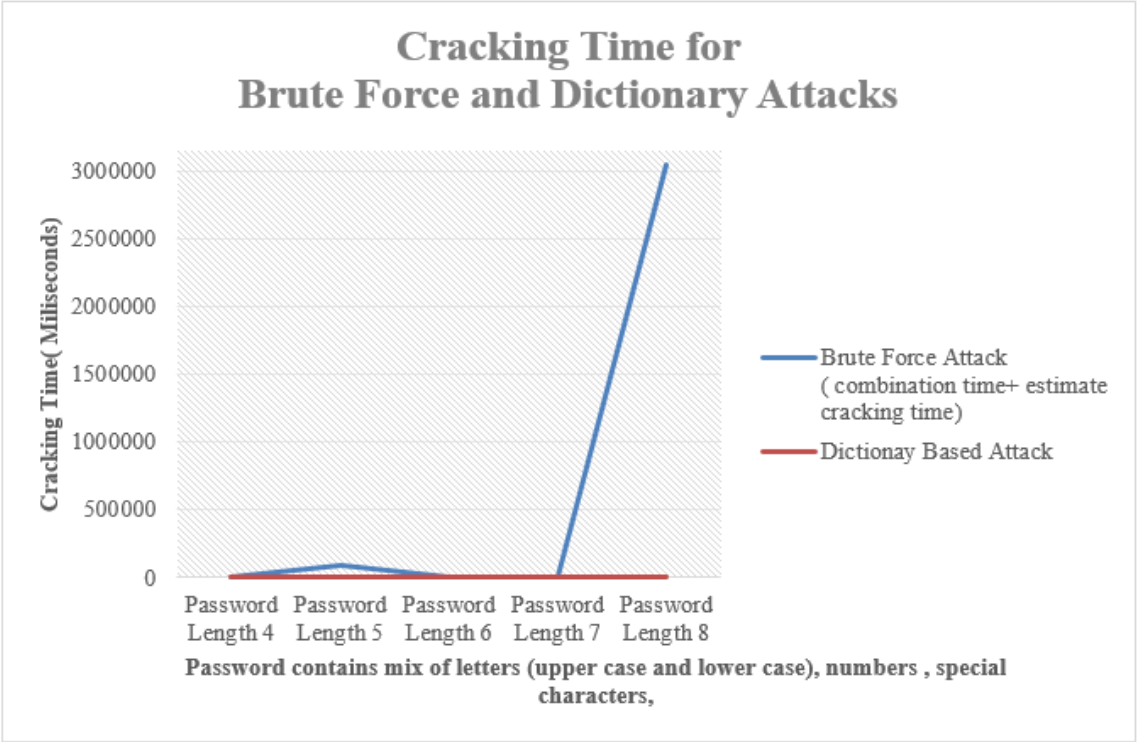


Figure 4.11 Line graph for Brute Force and Dictionary Based Attack crack time

CHAPTER 5
CONCLUSION, LIMITATIONS AND FURTHER EXTENSIONS

The proposed system has provided the secure authentication that is strong for brute force and dictionary attack. This system implements the Brute Force Attack by using combination algorithm, it is easy to implement and can try out all the possible combinations of ASCII character. It collects the dictionary words from the file given in crackstation.net to implement the Dictionary Attack. It uses these kinds of attacks as *white attack* to test the security of web application upon Broken Authentication. Therefore, this system can detect the vulnerability on web application for Broken Authentication by forcing only Brute force and Dictionary Attack.

5.1 Advantages

The proposed system was implemented for the detecting of broken authentication by using brute force and dictionary-based attack. This also describes the passwords that are used in this system are strong enough or not. There are also some equations for estimating cracking time based on the password length. What are the strong passwords and how to create the strong password and how to protect password from attackers? There are also ways to protect the password cracking attack by using multifactor authentication, one-time password and tokens etc.

5.2 Limitations and Further Extensions

Simple brute-force attack commonly uses automated tools to guess all possible passwords until the correct input is identified. This is an old but still effective attack method for cracking common passwords. The system can only crack the plain text by using simple brute force attack. The system will use the hashing algorithm in brute force attack in future study.

No hashing algorithm is 100% secure against brute force attacks. However, even with hardware-assisted password cracking (using the GPU to try passwords), the time it takes to crack a sufficiently long password is astronomical. If there has a password of eight characters, it could be vulnerable to a brute force attack. But if a few more characters are added, the time it takes to crack increases radically.

REFERENCES

[1] A. Christy Sathyabama University, Chennai, Tamilnadu, India, D. Saravanan Sathyabama University, Chennai, Tamilnadu, India, "An Analysis of Markov Password Against Brute Force Attack for Effective Web Applications", January 2014.

[2] ALBERT NJENHUIS and HERBERT S. WILF, "Combinatorial Algorithms For Computers and Calculators, Second Edition", pg-45-61, 1978.

[3] B. R. Heap, "Permutations by interchanges", The Computer Journal, Volume 6, Issue 3, November 1963, Pages 293–298 01 November 1963.

[4] Crack Word list text file from "https://crackstation.net/crackstation-wordlist-password-cracking-dictionary.htm"

[5] David C. Feldmeier and Philip R. Karn Bellcore, "UNIX Password Security-Ten Years Later", 445 South Street Morristown, NJ 07960.

[6] Konark Truptiben Dave, "Brute-force Attack "Seeking but Distressing", Department of Computer Engg. & Information Tech. C.U.Shah Technical Institute of Diploma Studies, Surendranagar-363001, Gujarat, India, International Journal of Innovations in Engineering and Technology (IJIET).

[7] L. Bošnjak, J. Sreš and B. Brumen, "Brute-force and dictionary attack on hashed real-world passwords", May 2018.

[8] OWASP Top Ten, [online] available: https://owasp.org/www-project-top-ten/, visit on February 2022.

[9] S. Vaithyasubramanian Sathyabama University, Chennai, Tamilnadu, India Satomi Saito Koji Maruhashi1 Masahiko Takenaka1 Satoru Torii1"TOPASE: Detection and Prevention of Brute Force Attacks with Displined IPs from IDs Logs, November, 2015.

[10] Signal Sciences, "Brute Force Attack Protection".

[11] Tobias Lundberg, "Comparison of Automated Password Guessing Strategies", Master of Science Thesis in Electrical Engineering Department of Electrical Engineering, Linköping University, 2019.