Experiment 06

Learning Objective: Implement Brute Force Attack

Tools: C/C++/Java/Python

Theory:

A brute force attack is a hacking method that uses trial and error to crack passwords, login credentials, and encryption keys. It is a simple yet reliable tactic for gaining unauthorized access to individual accounts and organizations' systems and networks. The hacker tries multiple usernames and passwords, often using a computer to test a wide range of combinations, until they find the correct login information.

The name "brute force" comes from attackers exhaustive attempts to gain access to user accounts. Despite being an old cyberattack method, brute force attacks are tried and tested and remain a popular tactic with hackers.

Types of Brute Force Attacks

There are various types of brute force attack methods that allow attackers to gain unauthorized access and steal user data.

1. Simple Brute Force Attacks

A simple brute force attack occurs when a hacker attempts to guess a user's login credentials manually without using any software. This is typically through standard password combinations or personal identification number (PIN) codes.

These attacks are simple because many people still use weak passwords, such as "password123" or "1234," or practice poor password etiquette, such as using the same password for multiple websites. Passwords can also be guessed by hackers that do minimal reconnaissance work to crack an individual's potential password.

2. Dictionary Attacks

A dictionary attack is a basic form of brute force hacking in which the attacker selects a target, then tests possible passwords against that individual's username. The attack method itself is not technically considered a brute force attack, but it can play an important role in a bad actor's password-cracking process.

The name "dictionary attack" comes from hackers running through dictionaries and amending words with special characters and numbers. This type of attack is typically time-consuming and has a low chance of success compared to newer, more effective attack methods.

3. Hybrid Brute Force Attacks

A hybrid brute force attack is when a hacker combines a dictionary attack method with a simple brute force attack. It begins with the hacker knowing a username, then carrying out a dictionary attack and simple brute force methods to discover an account login combination.

The attacker starts with a list of potential words, then experiments with character, letter, and number combinations to find the correct password. This approach allows hackers to discover passwords that combine common or popular words with numbers, years, or random characters, such as "Rover2020."

4. Reverse Brute Force Attacks

A reverse brute force attack sees an attacker begin the process with a known password, which is typically discovered through a network breach. They use that password to search for a matching login credential using lists of millions of usernames. Attackers may also use a commonly used weak password, such as "Password123," to search through a database of usernames for a match.

5. Credential Stuffing

Credential stuffing preys on users weak password etiquettes. Attackers collect username and password combinations they have stolen, which they then test on other websites to see if they can gain access to additional user accounts. This approach is successful if people use the same username and password combination or reuse passwords for various accounts and social media profiles.

Implementation

Source code implementing Brute Force Attack:

```
#include <stdio.h>
#include <string.h>

int check_password(char *str, char *target)
{
return strcmp(str, target) == 0;
}
```

```
void bruteForce(char *str, int index, int length, char *target)
{
const char alphabet[] =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!@#$%^&*()_-
+=<>?";

if (index == length)
{
if (check_password(str, target))
{
printf("Password found: %s\n", str);
}
return;
}

for (int i = 0; i < strlen(alphabet); i++)
{
str[index] = alphabet[i];
bruteForce(str, index + 1, length, target);
}
}

int main()
{
char target[10];
printf("Enter the target password: ");
scanf("%s", target);

int len;
printf("Enter the maximum length of the password: ");
scanf("%d", &len);

bruteForce(target, 0, len, target);

return 0;
}
```

Output:

Output of Implementation of Brute Force Attack:

```
            printf("Password found: %s\n", str);
        }
        return;
    }

    for (int i = 0; i < strlen(alphabet); i++)
    {
        str[index] = alphabet[i];
        bruteForce(str, index + 1, length, target);
    }
}

int main()
{
    char target[10];
    printf("Enter the target password: ");
    scanf("%s", target);

    int len;
    printf("Enter the maximum length of the password: ");
    scanf("%d", &len);

    bruteForce(target, 0, len, target);

    return 0;
}
```

input

```
Password found: aaaaaaad4&
Password found: aaaaaaad4*
Password found: aaaaaaad4(
Password found: aaaaaaad4)
Password found: aaaaaaad4_
Password found: aaaaaaad4-
Password found: aaaaaaad4+
Password found: aaaaaaad4=
Password found: aaaaaaad4<
Password found: aaaaaaad4>
Password found: aaaaaaad4?
Password found: aaaaaaad5a
```

Learning Outcomes: The student should have the ability to

LO1: Detect weak passwords

LO2: Understand Password Cracking Methods

LO3: Use Password Cracking Methods to implement mitigation strategies

Course Outcomes: Upon completion of the course students will be able to identify & usePassword Cracking Mechanisms to detect weak passwords.

Conclusion: In summary, a brute-force attack can be as simple as a sledgehammer or as nuanced as a lock pick set. Regardless of the method, the outcome remains the same—unauthorized access that can wreak havoc on personal and organizational levels.

Name: Aryan Tiwari

Class: SE-CSE

Roll.No: 57

For Faculty Use:

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				