Passwort Hacking: Vergleich von Brute-Force-Angriffen, Dictionary-Angriffen und Rainbow Tables zur Authentifizierungssicherheit

Seminararbeit

für die Prüfung zum

Bachelor of Science

des Studiengangs Informatik

an der

Dualen Hochschule Baden-Württemberg Lörrach

Marvin Obert

6. Juni 2023

Kurs	TIF21B
Ausbildungsfirma	VEGA Grieshaber KG, Schiltach
Wissenschaftlicher Betreuer	Prof. Dr. Hans Mustermann

Ehrenwörtliche Erklärung

Ich versichere hiermit, dass ich meine Bachelorarbeit (bzw. Projektarbeit oder Seminararbeit) mit dem Thema:

Passwort Hacking: Vergleich von Brute-Force-Angriffen, Dictionary-Angriffen und Rainbow Tables zur Authentifizierungssicherheit

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Lörrach, 15. Februar 2000

_____

Marvin Obert

Hinweis zum Umfang der Arbeit

Der Textteil der vorliegenden Arbeit - beginnend mit der Einleitung bis ausschließlich Quellenverzeichnis - umfasst 17 Seiten.

Kurzfassung

Hier beginnt die Kurzfassung ihrer wissenschaftlichen Arbeit…

Inhaltsverzeichnis

Abkürzungsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Einführung

> *„More companies are moving to new stronger technologies to authenticate user identities, like biometrics. Because it's just too easy for hackers to figure out usernames and passwords, like "password"" or 12345... 7. Those are some of my previous passwords. I've changed them since then."*
>
> Barack Obama[1]

In der heutigen Zeit sind Passwörter eine der grundlegendsten Methoden um Benutzer zu identifizieren. Allerdings ist die Komplexität der meisten Passwörter sehr gering und somit ist das Benutzernamen und Passwort erraten sehr leicht. Wie der ehemalige US-Präsident Barack Obama betont, haben Unternehmen erkannt, dass sie auf neue, stärkere Technologien umsteigen müssen, um die Identität von Benutzern zu authentifizieren, wie zum Beispiel biometrische Verfahren. Diese Authentifikationsmethoden sind angesicht der Passwörter des ehemalige Präsidenten ein deutlich stärkere und sichere Maßnahme.

1.2 Zieldefinition

Das Ziel dieser Seminararbeit besteht darin, eine umfassende Vergleichsanalyse ausgewählter Passwort-Hacking-Methoden durchzuführen. Dazu werden Brute-Force-Angriffe, Dictionary-Angriffe und Rainbow Tables im Hinblick auf Effizienz, Erfolgsrate und Ressourcenanforderungen untersucht. Zusätzlich erfolgt eine eingehende Untersuchung der Sicherheit des weit verbreiteten WLAN-Verschlüsselungsstandards WPA2 und der Diskussion möglicher Schutzmaßnahmen.

Um dieses Ziel zu erreichen, wird sowohl ein theoretischer Vergleich durchgeführt, bei dem eine Nutzwertanalyse erstellt wird, als auch ein praktischer Vergleich in einem konstruierten Testfall durchgeführt. Der praktische Vergleich ermöglicht eine Überprüfung der Wirksamkeit und Anwendbarkeit der verschiedenen Methoden.

[1] Nick Statt, 2015, Vgl.

Durch diese umfassende Herangehensweise soll ein fundiertes Verständnis für die Stärken und Schwächen der einzelnen Passwort-Hacking-Methoden gewonnen werden, um Empfehlungen für den Schutz vor solchen Angriffen abzuleiten. Die Kombination aus theoretischer Analyse und praktischer Anwendung ermöglicht eine ganzheitliche Betrachtung des Themas.

1.3 Methodik

1. Einführung der gängigen Hacking-Methoden: Als erster Schritt erfolgt eine umfassende Einführung in die verschiedenen gängigen Hacking-Methoden. Dabei werden die Methoden ausführlich erläutert, um den Leserinnen und Lesern einen fundierten Überblick über die unterschiedlichen Angriffstechniken zu geben.

2. Erklärung der Rainbow Tables: Im Anschluss daran erfolgt eine detaillierte Erklärung der Rainbow Tables, da diese für den praktischen Teil unverzichtbar sind. Es wird eine klare Definition der Rainbow Tables gegeben sowie ihre Funktionsweise und ihr Zweck im Kontext des Passwort-Hackings erläutert.

3. Aufstellung von Vergleichskriterien: Nach der Einführung und Erläuterung der Hacking-Methoden werden spezifische Vergleichskriterien aufgestellt, anhand derer die verschiedenen Methoden verglichen werden können. Dadurch wird eine strukturierte Grundlage für den Vergleich geschaffen.

4. Durchführung einer Nutzwertanalyse: Im nächsten Schritt erfolgt die Durchführung einer Nutzwertanalyse, um die verschiedenen Hacking-Methoden anhand der aufgestellten Kriterien zu bewerten. Dadurch können die Vor- und Nachteile jeder Methode objektiv beurteilt und eine Rangfolge erstellt werden.

5. Beschreibung der Testumgebung: Anschließend wird die Testumgebung detailliert beschrieben, die für die praktische Durchführung der Tests aufgesetzt wird. Dabei werden die verwendeten Hard- und Softwarekomponenten sowie die genauen Schritte und Vorgehensweisen für die Tests erläutert.

6. Praktische Durchführung der Tests: Die Tests werden gemäß der beschriebenen Testumgebung durchgeführt und die Ergebnisse sorgfältig dokumentiert. Es werden die angewandten Methoden, verwendeten Tools und erzielten Ergebnisse beschrieben. Dabei wird darauf geachtet, dass alle relevanten Informationen klar und präzise dargestellt werden.

7. Zusammenfassung der Ergebnisse: Am Ende der Methodik werden die Ergebnisse der praktischen Tests zusammengefasst und eventuelle Erkenntnisse oder Besonderheiten hervorgehoben. Es wird sichergestellt, dass die dokumentierten Ergebnisse den zuvor aufgestellten Vergleichskriterien entsprechen und eine Grundlage für die anschließende Diskussion bilden.

2 Grundlagen

2.1 Brute Force

2.1.1 Definition

Ein Brute-Force-Angriff bezeichnet den Versuch, ein Passwort, einen Benutzernamen oder einen Schlüssel zu knacken, indem systematisch alle möglichen Zeichenkombinationen ausprobiert werden. Dieser Ansatz basiert darauf, dass es für viele Probleme in der Informatik keine effizienten Algorithmen gibt. Daher stellt der Brute-Force-Angriff eine einfache Methode dar, um die Lösung zu finden, indem alle potenziellen Lösungen nacheinander ausprobiert werden.[2] Es ist eine zeitaufwändige Methode, da bei längeren oder komplexeren Passwörtern oder Schlüsseln eine große Anzahl von Kombinationen durchprobiert werden muss, um die richtige Lösung zu finden. Dennoch kann der Brute-Force-Angriff effektiv sein, insbesondere wenn die gesuchte Lösung nur eine begrenzte Anzahl von Möglichkeiten hat.

2.1.2 Funktionsweise

Die Funktionsweise eines Brute-Force Algorithmus lässt sich anhand des folgenden Code Beispiels erklären.

[2] Vgl. Wikipedia, 2023.

```
public string Bruteforce(string password, char[] symbols)
{
    List<char> bruteforcepassword = new List<char>();

    if (password == new string(bruteforcepassword.ToArray()))
    {
        return new string(bruteforcepassword.ToArray());
    }

    while (password != new string(bruteforcepassword.ToArray()))
    {
        int length = bruteforcepassword.Count;
        char lastchar = bruteforcepassword[length - 1];
        int indexofcharinsymbols = Array.IndexOf(symbols, lastchar);

        if (indexofcharinsymbols != length - 1)
        {
            bruteforcepassword[length - 1] = symbols[indexofcharinsymbols + 1];
        }
        else
        {
            int iterator = bruteforcepassword.Count - 1;
            while (iterator > -1 && bruteforcepassword[iterator] == symbols[symbols.Length - 1])
            {
                bruteforcepassword[iterator] = symbols[0];
                iterator--;
            }

            if (iterator == -1)
            {
                bruteforcepassword.Add(symbols[0]);
            }
        }
    }

    return new string(bruteforcepassword.ToArray());
}
```

Abbildung 1: selbstgeschriebener Quellcode eines Brute-Force Algorithmus[3]

In der Abbildung ist ein Beispiel Code zusehen, welcher ein Passwort anhand eines Brute-Force Algorithmuses findet. Der gegebene Code implementiert eine Funktion namens “Bruteforce“ in C-Sharp. Diese Funktion verwendet eine Brute-Force-Methode, um ein Passwort zu erraten, indem sie systematisch alle möglichen Kombinationen von Zeichen ausprobiert. Der Funktion werden zwei Parameter übergeben: das zu erratende Passwort als Zeichenfolge und ein Array von Symbolen, aus denen die Kombinationen gebildet werden sollen. Zu Beginn wird eine leere Liste namens “bruteforcepassword“ erstellt. Dann wird überprüft, ob das gegebene Passwort bereits mit der aktuellen Kombination übereinstimmt. Wenn dies der Fall ist, wird die aktuelle Kombination als Zeichenfolge zurückgegeben. Ansonsten wird eine Schleife gestartet, die solange läuft, bis das gegebene Passwort mit der aktuellen Kombination übereinstimmt. In jeder Iteration wird die Länge der aktuellen Kombination bestimmt und das letzte Zeichen abgerufen. Es wird der Index dieses Zeichens im Symbol-Array ermittelt. Wenn der Index nicht gleich der Länge minus eins ist, wird das letzte Zeichen durch das nächste Zeichen im Symbol-Array ersetzt. Wenn der Index gleich der Länge minus eins ist, bedeutet dies, dass das

letzte Zeichen bereits das letzte verfügbare Symbol ist. In diesem Fall wird ein Iterator initialisiert, der vom Ende der aktuellen Kombination bis zum Anfang durchläuft. Solange das aktuelle Zeichen das letzte verfügbare Symbol ist, wird es durch das erste Symbol im Symbol-Array ersetzt. Wenn der Iterator den Anfang erreicht und das aktuelle Zeichen immer noch das letzte Symbol ist, wird das erste Symbol am Ende der aktuellen Kombination hinzugefügt. Nachdem die Schleife beendet ist und das Passwort gefunden wurde, wird die aktuelle Kombination als Zeichenfolge zurückgegeben.

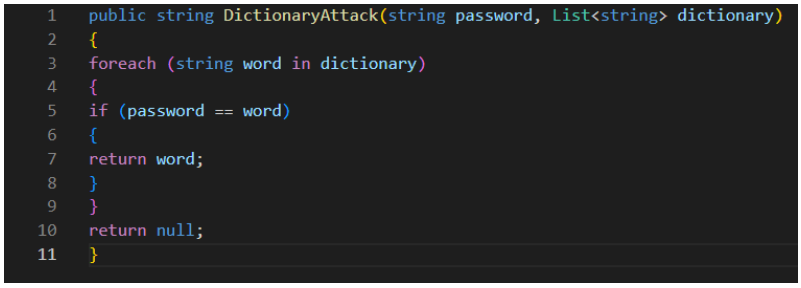
2.2 Dictionary Attacks

2.2.1 Defintion

Ein Dictionary-Angriff, auch als Wörterbuch-Angriff bezeichnet, ist eine Methode, um Passwörter oder Schlüssel durch systematisches Ausprobieren einer Liste häufig verwendeter Wörter, Phrasen oder Passwortkombinationen zu knacken. Bei einem Dictionary-Angriff wird eine vordefinierte Liste, auch als Wörterbuch oder Dictionary bezeichnet, verwendet, die potenzielle Passwörter oder Schlüssel enthält. Diese Liste kann verschiedene Formen annehmen, wie beispielsweise eine Sammlung häufig verwendeter Wörter, bekannte Passwörter, gebräuchliche Phrasen oder Kombinationen aus Wörtern und Zahlen. Der Angriff erfolgt, indem das Programm oder Skript die Liste der Wörter systematisch mit dem Ziel durchprobiert, das Passwort oder den Schlüssel zu finden. Es werden verschiedene Variationen und Kombinationen der Wörter aus dem Wörterbuch ausprobiert, einschließlich der Verwendung von Zahlen, Sonderzeichen und Groß- oder Kleinschreibung.

2.2.2 Funktionsweise

Die Abbildung veranschaulicht die Funktionsweise eines einfachen Dictionary-Attack-Algorithmus.



```
public string DictionaryAttack(string password, List<string> dictionary)
{
foreach (string word in dictionary)
{
if (password == word)
{
return word;
}
}
return null;
}
```

Abbildung 2: selbstgeschriebener Quellcode eines Dictionary Algorithmus[4]

In diesem Beispielcode wird eine Methode ”DictionaryAttack”implementiert, die ein Passwort und eine Liste von Wörtern als Parameter akzeptiert. Der Code durchläuft anschließend jedes Wort in der Liste und vergleicht es mit dem gegebenen Passwort. Wenn das Passwort mit einem Wort aus der Liste übereinstimmt, wird das gefundene Wort als Ergebnis zurückgegeben. Andernfalls wird null zurückgegeben, wenn keine Übereinstimmung gefunden wurde.

2.3 Komplexität von Passwörtern

Die Anzahl der möglichen Lösungen für ein Passwort kann durch die folgende Formel berechnet werden: Kombinationen = Zeichenanzahl $^{\text{Passwortlänge}}$[5]. Diese Formel ermöglicht die Untersuchung der verschiedenen Kombinationsmöglichkeiten von Passwörtern. Dabei werden die Ziffern betrachtet, die aus 10 verschiedenen Symbolen bestehen. Ebenso werden die Kleinbuchstaben der deutschen Sprache ohne Sonderzeichen berücksichtigt, die aus 26 Symbolen bestehen. Zusätzlich werden auch Kleinbuchstaben, Großbuchstaben und Ziffern betrachtet, wodurch sich eine Menge von 62 verschiedenen Symbolen ergibt.

[5]Vgl Bronstein/Semendjajew/Musiol/Muhlig, 1999, S. 2.

Passwortlänge	Ziffern	Kleinbuchstaben	Kleinbuchstaben, Großbuchstaben und Ziffern
1	10^1	26	62
2	10^2	676	3,844
3	10^3	17,576	238,328
4	10^4	456,976	14,776,336
5	10^5	11,881,376	916,132,832
6	10^6	308,915,776	56,800,235,584
7	10^7	8,031,810,176	3,521,614,606,208
8	10^8	208,827,064,576	218,340,105,584,896
9	10^9	5,429,503,678,976	13,537,086,546,263,552
10	10^{10}	141,167,095,653,376	839,299,365,868,340,224
11	10^{11}	3,670,344,486,987,776	52,031,252,847,222,976,512
12	10^{12}	95,428,956,661,682,176	3,226,266,762,397,899,821,824

Tabelle 1: Kombinationsmöglichkeiten von Passwörtern

Interessant ist die Passwortlänge von acht, da ein WPA2-Passwort mindestens aus acht Zeichen bestehen muss. Bei der Betrachtung der möglichen Kombinationen wird deutlich, dass bei Verwendung von nur Ziffern 10 Millionen Möglichkeiten existieren. Im Vergleich dazu gibt es mehr als das 2.000-fache an Möglichkeiten bei Verwendung von Kleinbuchstaben. Wenn wir die Kombination von Kleinbuchstaben, Großbuchstaben und Ziffern betrachten, ergibt sich ein Unterschied, der größer als der Faktor 1.000 ist.

2.4 WPA 2

WPA2 (Wi-Fi Protected Access 2) ist ein Sicherheitsprotokoll, das in Wi-Fi-Netzwerken verwendet wird, um die Vertraulichkeit und Integrität der drahtlosen Kommunikation zu gewährleisten. Es ist der Nachfolger von WPA und bietet eine stärkere Verschlüsselung und verbesserte Sicherheitsfunktionen. Die Funktionsweise von WPA2 basiert auf einem Vier-Wege-Handshake, der zwischen dem Client (z. B. ein Laptop oder ein Smartphone) und dem Access Point (der drahtlosen Basisstation) stattfindet. Der Handshake ermöglicht es beiden Parteien, sich gegenseitig zu authentifizieren und einen gemeinsamen geheimen Sitzungsschlüssel zu etablieren, der für die Verschlüsselung des Datenverkehrs verwendet wird. WPA2 verwendet zur Verschlüsselung des Passworts den PBKDF2 (Password-Based Key Derivation Function 2)-Algorithmus. Bei PBKDF2 handelt es sich um einen Algorithmus zur Derivation von Schlüsseln basierend auf einem Passwort.

Der Schlüssel für die Verschlüsselung setzt sich dabei aus verschiedenen Komponenten zusammen, nämlich dem Passwort selbst, dem Salt (einem zufälligen Wert), der Anzahl der Iterationen, der verwendeten Hash-Funktion und der Länge des abgeleiteten Schlüssels. Diese Parameter werden in der Formel key = (password, salt, iterations-count, hash-function, derived-key-len) festgelegt.[6]

2.5 Rainbow Tables

2.5.1 Definition

Die Rainbow-Tabelle, auch bekannt als Regenbogentabelle, ist eine Datenstruktur, die von Philippe Oechslin entwickelt wurde IT-Forensik Wiki, 2019. Sie ermöglicht eine effiziente und speichersparende Suche nach der ursprünglichen Zeichenfolge, in der Regel ein Passwort, basierend auf einem gegebenen Hashwert. Die Rainbow-Tabelle ist eine bedeutende technologische Entwicklung im Bereich der kryptografischen Sicherheit und spielt eine wichtige Rolle bei der Entschlüsselung von Passwörtern und der Sicherheitsanalyse von Hashfunktionen.

2.5.2 WPA2

Im Kontext von WPA2 werden Rainbow-Tabellen vor dem Angriff auf das WPA2-Netzwerk neu generiert. Beim Erzeugen des Schlüssels wird folgende Formel verwendet: key = pbkdf2(Passwort, SSID, 4096, HMAC-SHA1, 256). Die SSID des WLANs wird als Teil des Schlüssels verwendet, um eine eindeutige Verknüpfung zwischen dem Passwort und dem Netzwerk herzustellen. Vor dem Angriff können mit Hilfe von Techniken wie der Dictionary-Attack oder Brute-Force viele Schlüssel-Hashwert-Paare erzeugt werden. Im 4-Wege-Handshake, der zur Authentifizierung zwischen dem Access Point und dem Client stattfindet, wird das verhashte Passwort aufgezeichnet. In der dritten Nachricht des 4-Wege-Handshakes, dem EAPOL-Key, ist der PTK (Pairwise Transient Key) enthalten. Der PTK setzt sich aus dem PMK (Pairwise Master Key), zwei zufälligen Zahlen, die zuvor ausgetauscht wurden, sowie den

[6] Svetlin Nakov, 2022.

MAC-Adressen des Benutzers und des Access Points zusammen.[7] Der PMK ist das verhashte Passwort, das in der Rainbow-Tabelle gesucht werden kann. Wenn ein passender Hashwert in der Rainbow-Tabelle gefunden wird, bedeutet dies, dass das ursprüngliche Passwort ebenfalls gefunden wurde. Die Rainbow-Tabelle ermöglicht es Angreifern, auf effiziente Weise Passwörter zu entschlüsseln und somit Zugang zu einem geschützten WPA2-Netzwerk zu erlangen.

[7] admin, 2019, VGL.

3 Kriterienauswahl

3.1 Effizienz

Die Effizienz bezieht sich auf die Geschwindigkeit, mit der die Algorithmen Passwörter knacken können. Hierbei kann man die durchschnittliche Zeit oder die Anzahl der Versuche betrachten, die benötigt werden, um ein Passwort erfolgreich zu ermitteln. Ein effizienterer Algorithmus erzielt das gewünschte Ergebnis schneller.

3.2 Erfolgsquote

Die Erfolgsquote gibt an, wie häufig die Algorithmen in der Lage sind, das korrekte Passwort zu identifizieren. Hierbei ist es wichtig zu berücksichtigen, ob die Algorithmen auch komplexe Passwörter oder solche mit geringfügigen Variationen erfolgreich erkennen können. Eine höhere Erfolgsquote zeigt die Fähigkeit eines Algorithmus, verschiedene Passwortarten zu knacken.

3.3 Ressourcenbedarf

Der Ressourcenbedarf bezieht sich auf den benötigten Speicherplatz und die Rechenleistung, die die Algorithmen erfordern. Ein Algorithmus, der weniger Ressourcen benötigt, ist in der Regel effizienter und praktischer einzusetzen.

3.4 Anpassungsfähigkeit

Die Anpassungsfähigkeit bezieht sich darauf, wie gut die Algorithmen auf unterschiedliche Szenarien und Anforderungen angewendet werden können. Dies umfasst die Fähigkeit, verschiedene Wörterbücher oder Passwortlisten zu verwenden sowie die Möglichkeit, die Algorithmen an spezifische Passwortrichtlinien anzupassen.

4 Nutzwertanalyse

In der Nutzwertanalyse werden die Algorithmen in den verschiedenen Kriterien mit Werten zwischen 1 - 10 bewertet. Je höher die Zahl ist desto besser ist der Algorithmus.

4.1 Effizienz

Der Brute-Force-Algorithmus zeichnet sich durch seine Fähigkeit aus, alle möglichen Kombinationen zu überprüfen, um das gesuchte Passwort zu finden. Dieser umfassende Ansatz erfordert jedoch eine große Anzahl von Anfragen, um das gewünschte Ergebnis zu erzielen. Trotz dieses erhöhten Ressourcenbedarfs verfügt der Brute-Force-Algorithmus über eine strukturierte Vorgehensweise, die es ihm ermöglicht, relativ schnell neue potenzielle Kombinationen zu generieren, ohne dabei doppelte Anfragen zu senden. Es ist jedoch wichtig anzumerken, dass der Brute-Force-Algorithmus durch den Einsatz von Techniken wie der Einbindung von GPUs und der Bildung von Clustern effizienter gemacht werden kann. Diese Ansätze ermöglichen eine Parallelisierung der Berechnungen und eine schnellere Verarbeitung großer Mengen an Daten. Durch die Nutzung dieser Ressourcen kann der Brute-Force-Algorithmus die Zeit, die für das Durchprobieren aller möglichen Kombinationen benötigt wird, erheblich reduzieren In Bezug auf die Effizienz erhält der Brute-Force-Algorithmus eine Bewertung von 5 von 10 Punkten. Der Dictionary-Attack-Algorithmus nutzt eine Wortliste als Grundlage für seine Funktionsweise. Laut Statistiken sind rund 47 % der deutschen Benutzerpasswörter potenziell in einer solchen Wortliste enthalten.[8] Dies bedeutet, dass eine beträchtliche Anzahl von Passwörtern allein durch den Vergleich mit einer Wortliste ermittelt werden kann. Im Vergleich zum Brute-Force-Algorithmus erfordert der Dictionary-Attack nur wenige Anfragen, um die Überprüfung mit der Wortliste abzuschließen. Aufgrund der effizienten Ausnutzung der Wortliste und der schnellen Erkennung von passenden Kandidaten erhält der Dictionary-Attack-Algorithmus eine Bewertung von 8 von 10 Punkten in Bezug auf seine Effizienz.

[8] Web.de, 2021.

4.2 Erfolgsquote

Der Brute-Force-Algorithmus durchsucht systematisch alle möglichen Kombinationen, um das richtige Passwort zu finden. Da er keine Informationen über das Passwort verwendet, ist die Zeit, die benötigt wird, um das Passwort zu knacken, abhängig von der Länge und Komplexität des Passworts. Es gibt jedoch keine Garantie, dass das Passwort innerhalb einer bestimmten Zeit gefunden wird. In Bezug auf die Erfolgsquote erhält der Brute-Force-Algorithmus 10 von 10 Punkten, da er letztendlich jedes Passwort knacken kann. Der Dictionary-Attack-Algorithmus hingegen basiert auf einer vorgefertigten Liste von Wörtern, die mögliche Passwörter enthalten. Statistiken zeigen, dass etwa 47 % der deutschen Benutzer Passwörter verwenden, die in solchen Wortlisten enthalten sind. Dies bedeutet, dass mit einer geeigneten Wortliste ein beträchtlicher Anteil der Passwörter gefunden werden kann. Es ist jedoch wichtig zu beachten, dass diese Statistik auf eine spezifische Benutzergruppe zutrifft und nicht auf die gesamte Benutzerpopulation verallgemeinert werden kann. Unter Berücksichtigung von möglichen Passwörtern in den Kategorien “Sonstiges“ und “Keine Angaben“ könnte die Erfolgsquote des Dictionary-Attack-Algorithmus auf etwa 80 % steigen. Aufgrund der nicht garantierten Erfolgsquote werden dem Dictionary-Attack-Algorithmus 6 von 10 Punkten zugewiesen.

4.3 Ressourcenbedarf

Der Brute-Force-Algorithmus zeichnet sich durch seinen Ansatz aus, alle möglichen Kombinationen systematisch auszuprobieren, um das gesuchte Passwort zu finden. Dieser umfassende Ansatz erfordert jedoch eine erhebliche Menge an Rechenzeit, da alle potenziellen Kombinationen überprüft werden müssen. In Bezug auf den Ressourcenbedarf benötigt der Brute-Force-Algorithmus zwar nur wenig Speicher, da er das nächste Passwort anhand des vorherigen Passworts generieren kann, sogenannte Inkrementierung. Dennoch erhält er in diesem Kriterium eine Bewertung von 3 von 10 Punkten, da die Rechenzeit aufgrund der großen Anzahl von Kombinationen hoch ist.

Der Dictionary-Attack-Algorithmus hingegen benötigt während der Laufzeit eine moderatere Menge an Rechenzeit. Dies liegt daran, dass nur geringfügige Änderungen an

den bereits generierten Wörtern vorgenommen werden müssen, um weitere Kandidaten zu testen. Der eigentliche Vergleich der Wörter aus der Wortliste mit dem Passwort erfordert ebenfalls nur wenig Rechenleistung. Allerdings benötigt dieser Algorithmus einen höheren Speicherbedarf, da sowohl die bereits getesteten Wörter als auch die Wortliste gespeichert werden müssen. In Bezug auf den Ressourcenbedarf erhält der Dictionary-Attack-Algorithmus eine Bewertung von 6 von 10 Punkten, da der Rechenaufwand moderat ist, jedoch der Speicherbedarf höher ist.

4.4 Anpassungsfähigkeit

Der Brute-Force-Algorithmus zeichnet sich durch seine hohe Anpassungsfähigkeit aus, da verschiedene Symbole konfiguriert werden können. Diese Flexibilität ermöglicht es, Abwandlungen des Brute-Force-Algorithmus, sogenannte Masked Attacks, einzusetzen, bei denen gezielt nach spezifischen Mustern gesucht wird. Diese Anpassungsfähigkeit erlaubt eine präzisere und effizientere Suche nach dem Passwort. In Bezug auf die Anpassungsfähigkeit und die Fähigkeit, detaillierte Muster zu berücksichtigen, erhält der Brute-Force-Algorithmus eine Bewertung von 10 von 10 Punkten, da keine negativen Aspekte bekannt sind.

Bei Dictionary-Attacks besteht ebenfalls die Möglichkeit, sich durch den Einsatz verschiedener vorgefertigter Wortlisten an verschiedene Passwortrichtlinien anzupassen. Zudem können eigene Wortlisten für eine komplexere Anpassung generiert werden. Es ist jedoch wichtig anzumerken, dass die Suche nach passenden Wortlisten bei sehr komplexen und ungewöhnlichen Passwortrichtlinien eine Herausforderung darstellen kann. In solchen Fällen kann die Effektivität von Dictionary-Attacks eingeschränkt sein. Daher erhält der Dictionary-Attack in Bezug auf die Anpassung an spezielle Passwortrichtlinien eine Bewertung von 6 von 10 Punkten. Es ist jedoch zu beachten, dass für die meisten gängigen Passwortrichtlinien bereits umfangreiche und große Wortlisten vorhanden sind, was die Effektivität des Angriffs verbessert.

4.5 Gewichtung der Kriterien

1. Erfolgsquote (30%): Die Erfolgsquote ist ein entscheidendes Kriterium, da sie angibt, wie erfolgreich der Algorithmus beim Knacken von Passwörtern ist. Ein Algorithmus mit einer höheren Erfolgsquote sollte eine höhere Gewichtung erhalten, da dies seine Effektivität widerspiegelt.

2. Effizienz (30%): Die Effizienz betrifft die Zeit- und Ressourceneffizienz der Algorithmen. Ein effizienter Algorithmus benötigt weniger Ressourcen und Zeit, um das gewünschte Ergebnis zu erzielen. Daher ist die Effizienz ein wichtiges Kriterium, das in die Bewertung einfließen sollte.

3. Ressourcenbedarf (20%): Der Ressourcenbedarf betrifft den Einsatz von Rechenressourcen wie CPU, Speicher und Energie. Ein Algorithmus, der weniger Ressourcen benötigt, wird als vorteilhafter angesehen, da er kostengünstiger und effizienter ist.

4. Anpassungsfähigkeit (20%): Die Anpassungsfähigkeit bezieht sich auf die Fähigkeit der Algorithmen, sich an verschiedene Passwortrichtlinien anzupassen und spezifische Anforderungen zu erfüllen. Eine höhere Anpassungsfähigkeit ermöglicht es dem Algorithmus, effektiver auf verschiedene Szenarien zu reagieren und eine größere Bandbreite von Passwörtern zu knacken.

4.6 Fazit

Unter Berücksichtigung der Gewichtung erhält der Brute-Force-Algorithmus eine Gesamtpunktzahl von 7,1, während der Dictionary-Attack-Algorithmus eine Punktzahl von 6,6 erhält. Die Gewichtung der Kriterien spiegelt die Prioritäten und Anforderungen der Nutzwertanalyse wider. Der Dictionary-Attack-Algorithmus zeigt seine Stärken in seiner Effizienz und Flexibilität bei der Verwendung von vorgefertigten Wortlisten und begrenzten Ressourcen, um Passwörter zu knacken. Es ist ein schneller und einfacher Ansatz, der sich gut für Situationen eignet, in denen ein Passwort schnell gefunden werden soll. Allerdings besteht die Möglichkeit, dass der Algorithmus kein passendes Passwort findet. Im

Gegensatz dazu verbraucht der Brute-Force-Algorithmus deutlich mehr Ressourcen und ist ineffizienter, da er alle möglichen Kombinationen ausprobiert. Dennoch bietet er die Garantie, dass er letztendlich das gesuchte Passwort findet. Der Brute-Force-Algorithmus eignet sich daher für Situationen, in denen das Knacken eines Passworts absolut erforderlich ist. Ein metaphorischer Vergleich könnte sein, dass Dictionary-Attacks als “Universal-Schlüssel“ von bekannten Herstellern angesehen werden können, während Brute-Force-Attacken mit einem “Vorschlaghammer“ vergleichbar sind.

5 Aufsetzten der Umgebung

Für die Durchführung der Tests wurde die neueste Version von Kali Linux als Live-Distribution verwendet, die mithilfe des Tools Rufus auf einem USB-Stick bootfähig installiert wurde. Kali Linux ist eine renommierte Linux-Distribution, die speziell für Penetrationstests und Sicherheitstests entwickelt wurde, und ermöglicht den Zugriff auf aktuelle Sicherheitsfunktionen und Updates. Im ersten Test wurde auf einem Windows-Laptop ein WLAN-Hotspot eingerichtet, wobei als Sicherheitsstandard WPA2 gewählt wurde. Dieser Standard gilt als sicher und wird für drahtlose Netzwerke empfohlen. Bei der Auswahl der Passwortlänge für die Tests wurde eine Statistik herangezogen, die ergeben hat, dass 49% der Passwörter eine Länge von bis zu 10 Zeichen aufweisen.[9] Aufgrund dieser Erkenntnis wurden für die Tests Passwörter mit einer Länge von genau 10 Zeichen verwendet, um realitätsnahe Szenarien abzubilden. Insgesamt wurden die Tests unter Verwendung des Kali Linux-Images, das als Live-Distribution auf einem bootfähigen USB-Stick installiert wurde, durchgeführt. Dabei wurde ein WLAN-Hotspot mit WPA2 auf einem Windows-Laptop erstellt, während die Auswahl der Passwortlänge auf statistischen Daten basierte. Diese Vorgehensweise gewährleistet eine fundierte und wissenschaftlich solide Durchführung der Tests. Als Programm für die Brute-Force und Dictionary-Attacken wurde Hashcat verwendet. Hashcat ist ein leistungsstarker Open-Source-Tool zum Cracken von Passwörtern und zur Wiederherstellung von verschlüsselten Hashes. Um im folgenden auch noch Rainbow Table Attacken durchzuführen wurde rainbowcrack verwendet. RainbowCrack ist ein leistungsstarkes Tool zur Erzeugung und Verwendung von Rainbow Tables für die Passwortentschlüsselung. Für das Aufnehmen des Wlan Traffics wird hcxtools verwendet. Dieses ist für hashcat optimiert.

6 Durchführung der Test

[9] Web.de, 2022, Vgl.

Quellenverzeichnis

Internetquellen

admin (2019). *4-WAY HANDSHAKE*. URL: https://www.wifi-professionals.com/2019/01/4-way-handshake (besucht am 02. 06. 2023).

Bronstein/Semendjajew/Musiol/Muhlig (1999). *Kombinatorik kurz und knapp*. URL: https://stat.ethz.ch/education/semesters/WS_2005_06/info/KombinatorikZsfsg.pdf (besucht am 02. 06. 2023).

IT-Forensik Wiki (2019). *Rainbow Table*. URL: https://it-forensik.fiw.hs-wismar.de/index.php/Rainbow_Table (besucht am 02. 06. 2023).

Nick Statt (2015). *Obama, serious about cybersecurity, also delivers laughs*. URL: https://www.cnet.com/news/privacy/obama-serious-about-cybersecurity-also-delivers-laughs/ (besucht am 02. 06. 2023).

Svetlin Nakov (2022). *PBKDF2*. URL: https://cryptobook.nakov.com/mac-and-key-derivation/pbkdf2 (besucht am 02. 06. 2023).

Web.de (2021). *Wie erstellen/generieren Sie Ihr Passwort?* URL: https://de.statista.com/statistik/daten/studie/818733/umfrage/methoden-der-passworterstellung-in-deutschland/ (besucht am 02. 06. 2023).

Web.de (2022). *Wie lang sind Ihre am häufigsten verwendeten Passwörter?* URL: https://de.statista.com/statistik/daten/studie/988439/umfrage/laenge-von-passwoertern-in-deutschland/ (besucht am 02. 06. 2023).

Wikipedia (2023). *Brute-Force-Methode*. URL: https://de.wikipedia.org/wiki/Brute-Force-Methode (besucht am 03. 06. 2023).

Anhang

1. Digitale Version der Arbeit

2. Interviews

    2.1. Expertmann 2018