AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ INFORMATYKI

INSTYTUT INFORMATYKI

PRACA MAGISTERSKA

Optymalna generacja słowników dla kryptoanalizy słownikowej

Optimal dictionary generation for dictionary cryptoanalysis

Author:	**Mariusz Kadziela**
Field of study:	**Computer Science**
Thesis supervisor:	**dr hab. inż. Paweł Topa**

Kraków, 2024

Contents

1 Introduction

The ever-growing influence of digital technologies and the widespread integration of computer systems have ushered in unprecedented challenges in safeguarding sensitive data. Passwords, serving as a linchpin in securing information, are frequently stored as cryptographic hashes, rendering them irreversible. However, the persistent specter of cyber threats remains, with one particularly pervasive attack vector being the dictionary attack. This method systematically tests an extensive set of potential candidates against stored hash values in an attempt to unveil passwords [7].

This master's thesis embarks on a journey into the realm of dictionary-based cryptanalysis, placing a specific emphasis on optimizing the process of generating dictionaries tailored for such attacks. Drawing inspiration from foundational works in cryptography, such as *Understanding Cryptography: A Textbook for Students and Practitioners* by Christof Paar and Jan Pelzl [4], the primary objective of this research is to propose and evaluate efficient methods for constructing password dictionaries. These methods leverage information extracted from forensic evidence or any available data source to enhance the effectiveness of dictionary attacks. Concurrently, the thesis explores innovative approaches for password generation that draw upon established rules and patterns, reminiscent of those used by contemporary password-cracking algorithms [8, 5].

The scope of this thesis encompasses a comprehensive examination of various techniques and strategies for dictionary generation. We will scrutinize the intricacies of this endeavor and assess the performance of different methodologies, integrating insights from *Cryptography and Network Security* by William Stallings [7] and *Understanding Cryptography* by Christof Paar and Jan Pelzl [4]. The evaluation will take place within the context of recovering passwords from forensic material or fortifying defenses against password-based attacks in computer systems.

The significance of this research lies in its potential to fortify password security mechanisms, empowered by the knowledge distilled from *Understanding Cryptography* [4] and *Cryptography and Network Security* [7]. Additionally, it addresses the growing need for efficient password recovery techniques, particularly in cases involving law enforcement investigations, where optimizing the selection of potential passwords based on evidence is essential.

In the subsequent chapters, we will delve deeper into the theoretical foundations, methodologies, experiments, and results, striving to provide valuable insights into the development of optimized password dictionaries for the field of dictionary-based cryptanalysis.

2 Existing Dictionary-Based Cryptanalysis Methods

Dictionary-based cryptanalysis is a pivotal area of study in the cybersecurity domain, concentrating on endeavors to compromise passwords stored in the guise of cryptographic hashes. This technique involves assessing potential passwords enumerated in a dictionary to identify one generating a specific hash. In this discourse, we will present an overview of diverse dictionary-based cryptanalysis methods and furnish a detailed exposition of each.

2.1 Brute Force Attack

The Brute Force Attack methodology epitomizes the most simplistic and unambiguous approach to dictionary-based cryptanalysis [7]. In this approach, assailants employ a systematic method by testing every conceivable character combination to unveil the target password. This systematic approach commences with the shortest conceivable passwords and progressively advances to passwords of the desired length. While this technique is conceptually straightforward, it can be exceedingly time-consuming, particularly for extensive passwords. Nevertheless, it assures eventual discovery of the password if it resides within the attacker's dictionary.

2.1.1 Methodology

The methodology behind a Brute Force Attack is ostensibly straightforward but computationally intensive:

1. **Character Set Enumeration**: Attackers initiate by discerning the character set likely constituting the password, encompassing lowercase and uppercase letters, numbers, and special symbols. The attacker's dictionary is subsequently populated with these characters.

2. **Password Length Enumeration**: Attackers systematically generate passwords of assorted lengths, starting with single-character passwords and progressively increasing the length with each iteration. For instance, the attack may commence with 'a,' 'b,' 'c,' and so forth, then proceed to 'aa,' 'ab,' 'ac,' and so forth.

3. **Testing Passwords**: Each generated password candidate is tested by applying a cryptographic hash function to generate a hash value. This hash value is then compared to the target hash value (associated with the unknown password). If a match is found, the attacker has successfully discerned the password.

4. **Iterative Process**: This process endures until the attacker either depletes all conceivable combinations or successfully identifies the password.

2.1.2 Time and Resource Considerations

The primary drawback of the Brute Force Attack methodology is its computational intensity and temporal demands. The time required to test all conceivable character combinations grows exponentially with password length and complexity. For extensive and intricate passwords, a Brute Force Attack may be impractical, demanding an implausible amount of time to complete.

To redress this challenge, attackers frequently deploy various strategies to expedite the process, such as:

- **Precomputed Tables**: Leveraging precomputed tables like rainbow tables to diminish the time required for hash computations. - **Parallel Processing**: Employing multiple computational resources or distributed systems to test passwords concurrently. - **Password Complexity**: Prioritizing testing of passwords more likely to be selected by users, based on common patterns, dictionary words, or known substitutions.

2.1.3 Effectiveness and Limitations

The efficacy of a Brute Force Attack hinges largely on the complexity of the target password. While it assures eventual success, it may require an impractical amount of time for intricate and lengthy passwords. Conversely, it is highly effective against succinct and uncomplicated passwords.

Defenders frequently counter Brute Force Attacks by instituting security measures such as account lockouts, rate limiting, and the stipulation for robust and intricate passwords.

In summation, the Brute Force Attack methodology is a fundamental approach to dictionary-based cryptanalysis. It systematically tests every conceivable character combination to discover passwords, yet its efficacy fluctuates contingent on password complexity and length. Grasping this methodology is indispensable for both assailants and defenders in the realm of cybersecurity.

2.2 Dictionary Attack

The Dictionary Attack is a method extensively employed in dictionary-based cryptanalysis [7, 8]. In this approach, assailants utilize a pre-prepared dictionary or wordlist containing a myriad of potential passwords. Each password from the dictionary is systematically tested to ascertain if it corresponds to the target hash. The efficacy of the Dictionary Attack depends on the quality and comprehensiveness of the dictionary itself and whether the password is present within the dataset.

2.2.1 Methodology

The methodology of a Dictionary Attack can be delineated into the following steps:

1. **Dictionary Compilation**: Attackers commence by compiling a dictionary, also known as a wordlist. This dictionary encompasses a vast collection of potential passwords, including common words, phrases, previously breached passwords, keyboard patterns, and variations of known passwords.

2. **Hash Comparison**: The attacker systematically tests each password from the dictionary by applying a cryptographic hash function. The generated hash value is then compared to the target hash value (associated with the unknown password). If a match is found, the attacker successfully discerns the password.

3. **Testing Variations**: To augment their chances of success, attackers often test variations of each word in the dictionary, including numbers, symbols, or character substitutions.

4. **Iteration**: The process iteratively continues until the attacker exhausts the entire dictionary or successfully identifies the password.

2.2.2 Quality of the Dictionary

The efficacy of a Dictionary Attack hinges on the quality and comprehensiveness of the dictionary being used [4, 8]. A well-constructed dictionary may encompass:

- Common words and phrases. - Previously breached passwords from data leaks. - Keyboard patterns (e.g., "qwerty" or "123456"). - L33t speak variations (e.g., "pa

$$w0rd"). - Variations of known passwords (e.g., "Password1").$$

An extensive and diverse dictionary heightens the likelihood of successfully cracking passwords. Conversely, a limited or outdated dictionary diminishes the chances of success.

2.2.3 Effectiveness and Limitations

The effectiveness of a Dictionary Attack relies on several factors, including the quality of the dictionary, the complexity of the target password, and whether the password is present within the dataset [8]. This method is highly efficient when dealing with weak and commonly used passwords but becomes less effective against strong, unique, or randomly generated passwords.

Defenders often counter Dictionary Attacks by instituting password policies that advocate for strong, complex passwords. Additionally, monitoring login attempts and blocking accounts after multiple failed login trials can mitigate the impact of such attacks.

In summary, the Dictionary Attack method is a foundational approach to dictionary-based cryptanalysis. Its efficacy hinges on the quality of the dictionary and the characteristics of the target password. A profound understanding of this method is crucial for both attackers and defenders in the realm of cybersecurity.

2.3 Rainbow Table Attack

The Rainbow Table Attack is an advanced method of dictionary-based cryptanalysis that leverages a specialized data structure known as a rainbow table [7, 1]. This approach significantly expedites the password recovery process by transforming and storing hash values derived from various hash functions. During the attack, adversaries aim to find a hash in the rainbow table and subsequently retrieve the corresponding password.

2.3.1 Rainbow Tables: A Brief Overview

A rainbow table is a meticulously constructed data structure used to optimize the process of reversing hashed values [1]. These tables are designed to store precomputed hash chains, consisting of multiple hash values obtained through iterations of different hash functions. By generating these chains in advance, rainbow tables facilitate rapid hash value lookups and password recovery.

2.3.2 Methodology

The Rainbow Table Attack method follows these key steps:

1. **Table Generation**: Attackers commence by creating a rainbow table. This table contains a vast number of hash chains, each comprising multiple hash values. These chains are generated by iteratively applying various hash functions to initial values (known as "start points") and storing the resulting hash values.

2. **Hash Transformation**: The attacker applies the same hash functions used to create the table to the target hash value to transform it. This transformation yields a value that can be compared against the hash values stored in the rainbow table.

3. **Lookup in the Table**: The attacker searches the rainbow table for a match between the transformed target hash and the stored hash values. If a match is found, the attacker retrieves the corresponding password associated with the start point of that hash chain.

4. **Successive Reduction**: If no match is found initially, the attacker may employ a technique called "successive reduction." This involves iteratively applying a reduction function to the transformed target hash and comparing the resulting values to the stored hashes in the rainbow table until a match is located or the search is exhausted.

2.3.3 Effectiveness and Limitations

The Rainbow Table Attack is highly effective against hashed passwords, particularly when the rainbow table is well-constructed and comprehensive [1]. It excels in situations where attackers have access to precomputed rainbow tables, significantly reducing the time required to crack passwords.

However, this method has limitations:

- **Storage Requirements**: Generating and storing rainbow tables for all possible hash values and password combinations can be resource-intensive. - **Salted Hashes**: Rainbow tables are less effective against salted hashes, where a unique salt value is added to each password before hashing. - **Large Password Spaces**: As password complexity and length increase, the feasibility of constructing and using rainbow tables diminishes.

To counter Rainbow Table Attacks, defenders often implement salting to protect passwords and employ other measures to increase the complexity of password hashes.

In summary, the Rainbow Table Attack is a potent technique in dictionary-based cryptanalysis, leveraging precomputed hash chains to expedite password recovery. Its effectiveness depends on the quality of the rainbow table, and defenders should consider its limitations, such as salted hashes and large password spaces.

2.4 Rule-Based Techniques

Rule-Based Techniques constitute a category of dictionary-based cryptanalysis methods that rely on predefined rules and patterns commonly observed in password generation [7, 1]. These rules are designed to emulate known patterns, including dictionary words, keyboard sequences, character substitutions, and popular password patterns. The primary

objective of Rule-Based Techniques is to generate passwords that adhere to these patterns, thereby making them susceptible to being guessed by attackers.

2.4.1 Methodology

The methodology behind Rule-Based Techniques involves:

1. **Rule Sets**: Attackers utilize predefined rule sets that encompass specific patterns, transformations, and substitutions to apply during password generation [4, 1].

2. **Candidate Generation**: Password candidates are systematically generated based on the rules and patterns specified in the rule set. These candidates are added to the attacker's dictionary for use in dictionary attacks.

3. **Testing Candidates**: Each generated password candidate is tested against the target hash value. If a match is found, the attacker has successfully cracked the password.

4. **Iteration**: The process iterates until either the attacker exhausts the rule-based candidates or successfully identifies the password.

2.4.2 Strengths and Limitations

Strengths:

- **Efficiency**: Rule-Based Techniques efficiently generate passwords based on known patterns, making them effective for cracking passwords that conform to common composition rules. - **Speed**: Password generation using rules is relatively fast compared to some other cryptanalysis methods. - **Adaptability**: Rule sets can be customized to target specific datasets or password policies [4].

Limitations:

- **Predictability**: The predictability of rule-based passwords is a limitation, as defenders can employ countermeasures to detect and block such attacks. - **Limited for Complex Passwords**: More complex or unique passwords may not be cracked using rule sets alone, especially if they do not conform to common patterns. - **Salted Hashes**: Rule-Based Techniques are less effective against salted hashes, where a unique salt value is added to each password before hashing [1].

In conclusion, Rule-Based Techniques offer an efficient approach to dictionary-based cryptanalysis by generating passwords based on predefined rules and patterns. While they excel at cracking passwords that adhere to common composition rules, they may fall short when dealing with more complex or unique passwords that do not conform to these patterns. Understanding the strengths and limitations of Rule-Based Techniques is crucial for both attackers and defenders in the field of cybersecurity, especially in the context of a master's thesis [7].

2.5 Statistical-Based Attack

The Statistical-Based Attack is a method of dictionary-based cryptanalysis that deploys statistical analysis techniques to enhance the efficiency of password cracking [6, 1, 8, 5]. Instead of relying solely on predefined dictionaries, this approach utilizes statistical insights into password composition to generate more efficient and targeted dictionaries. It

takes into account factors such as letter frequency, special character usage, and other elements commonly found in passwords, with the goal of prioritizing the most likely password combinations.

2.5.1 Methodology

The methodology behind a Statistical-Based Attack is as follows:

1. **Data Collection**: Attackers begin by collecting a substantial dataset of passwords, which may include breached passwords, password dumps, or other sources of password information. This dataset serves as the basis for statistical analysis.

2. **Statistical Analysis**: Sophisticated statistical algorithms are applied to the collected password dataset [**?**]. These algorithms aim to identify patterns, trends, and common characteristics in passwords. Statistical analysis may include examining the frequency of letters, numbers, special characters, and their combinations.

3. **Dictionary Generation**: Based on the statistical insights gained from the analysis, attackers create customized dictionaries or wordlists. These dictionaries prioritize the most likely password combinations and patterns discovered during the statistical analysis. For example, if the analysis reveals that a significant portion of users use passwords like "P@ssw0rd," these patterns will be included in the dictionary.

4. **Dictionary Usage**: Attackers then use the generated dictionaries in dictionary attacks. These dictionaries are tailored to maximize the chances of success by focusing on the statistically prevalent password patterns.

2.5.2 Effectiveness and Limitations

The Statistical-Based Attack method can be highly effective in password cracking, particularly when attackers have access to comprehensive datasets and advanced statistical tools [3, 2]. By prioritizing the most common and statistically likely password combinations, this approach can yield quick results.

However, there are limitations to consider:

- **Dataset Quality**: The effectiveness of the attack depends on the quality and representativeness of the password dataset used for statistical analysis. Incomplete or biased datasets may lead to less accurate results. - **Evolving Passwords**: As users become more aware of password security, password composition habits change. Statistical-based dictionaries may become less effective against users who adopt complex, unique, or random passwords. - **Variability**: Password policies and requirements vary across different systems and platforms, making it challenging to create universally applicable statistical models.

To counter Statistical-Based Attacks, defenders often promote password policies that encourage users to create unique, complex, and unpredictable passwords. Additionally, monitoring and detecting abnormal login attempts can help thwart these attacks.

In summary, the Statistical-Based Attack method leverages statistical analysis to create tailored dictionaries focused on likely password patterns. Its effectiveness relies on the quality of the dataset and may be influenced by evolving password composition trends.

2.6 Summary

This article delves into the realm of Rule-Based Techniques in dictionary-based cryptanalysis, offering insights into their pivotal role in the cybersecurity landscape. These techniques wield substantial influence, guiding the actions of both cyber attackers and defenders. A nuanced understanding of these methods emerges as a critical asset for cybersecurity professionals dedicated to fortifying their systems against dictionary attacks. Simultaneously, it serves as an essential resource for those seeking to enhance their offensive capabilities.

3 Rule-Based Techniques in Dictionary-Based Cryptanalysis

In the realm of dictionary-based cryptanalysis, Rule-Based Techniques play a crucial role in generating potential passwords for testing against hashed values. These techniques rely on established rules and patterns to create a set of candidate passwords that attackers can use in their attempts to crack hashed passwords. In this chapter, we will delve into the intricacies of Rule-Based Techniques, explaining what they are, how they work, and their significance in the field of cybersecurity.

3.1 What Are Rule-Based Techniques? [7, 4]

Rule-Based Techniques, also known as password cracking rules, are predefined guidelines used in dictionary-based cryptanalysis. They are designed to mimic common user behaviors when creating passwords. These techniques encompass patterns based on dictionary words, keyboard sequences, substitutions, and other predictable characteristics.

The primary purpose of Rule-Based Techniques is to generate candidate passwords efficiently. These candidates are derived by systematically applying predefined rules to a base set of words or patterns. This process creates a comprehensive list of potential passwords for use in dictionary attacks.

These techniques are effective in cracking passwords that adhere to common patterns and predictable substitutions. Users often choose passwords that are easily memorable, but this predictability can be exploited by attackers using Rule-Based Techniques.

In summary, Rule-Based Techniques play a crucial role in dictionary-based password cracking by simulating user behavior and generating a wide range of potential password candidates.

3.2 How Rule-Based Techniques Work [7, 4]

The operation of Rule-Based Techniques can be broken down into several key steps:

1. **Rule Set Definition** [7]: A set of rules is defined, specifying the patterns and transformations to be applied to create candidate passwords. These rules can include appending numbers, adding special characters, or substituting letters with similar-looking characters.

2. **Rule Application** [7]: The rules are systematically applied to a base set of words, often derived from a dictionary or a list of common passwords. Each rule generates variations of the base words based on the defined transformations.

3. **Candidate Password Generation** [4]: The combinations of base words and rule-generated variations form a candidate password list. This list becomes part of the attacker's dictionary for password cracking attempts.

4. **Password Testing** [7]: Attackers use the generated candidate passwords to test them against hashed values in an attempt to find a match. If a match is found, the attacker has successfully cracked the password.

3.3 Common Rule-Based Techniques [7, 4, 8, 5]

Rule-Based Techniques encompass a variety of rules and patterns that are widely employed in password cracking attempts. These techniques leverage the predictability of human behavior when creating passwords. Below are some of the most commonly used rule-based strategies:

3.3.1 Dictionary-Based Rules [7]

Dictionary-based rules involve manipulating dictionary words by appending, prepending, or substituting characters to create potential passwords. These rules exploit the fact that many users base their passwords on easily memorable words or phrases. For instance:

- Appending numbers or special characters: "password123" or "secret!"
- Substituting characters: "pa$$w0rd" or "p@ssw0rd"
- Combining words: "sunflowerhouse" or "applejuice1"

Dictionary-based rules can significantly expand the pool of candidate passwords, as attackers explore various combinations and modifications of common words.

3.3.2 Keyboard Patterns [7]

Many users create passwords based on keyboard patterns due to their convenience. Attackers often employ rules that mimic these patterns, such as:

- Sequential key combinations: "qwerty" or "asdfgh"
- Keyboard walks: "zxcvbn" or "poiuyt"

These patterns are easy to type, but they are also easily guessable by attackers using Rule-Based Techniques. Recognizing and defending against such patterns is vital for password security.

3.3.3 L33t Speak [7]

L33t speak, or "leet speak," involves replacing letters with similar-looking characters or numbers. For example:

- Replacing 'e' with '3': "password" becomes "passw0rd"
- Substituting 'a' with '@': "apple" becomes "@pple"

L33t speak rules are effective because they create variations that resemble the original words while adding complexity. Users may think they have strong passwords, but attackers recognize these patterns and exploit them.

3.3.4 Common Patterns [7]

These rules target common password patterns that users frequently employ:

- Repeated characters: "aaa" or "1234"
- Common sequences: "abcd" or "8765"
- Well-known words: "admin" or "password123"

These patterns are predictable and often used by individuals who prioritize ease of memorization over security. Attackers can quickly identify and test these patterns in their dictionary-based attacks.

3.3.5 Combining Rules [7]

Attackers often combine multiple rules to generate a vast number of candidate passwords. For instance, they may apply dictionary-based rules first and then apply L33t speak to the results. This combination approach increases the chances of success in cracking passwords.

3.3.6 Defensive Strategies [7]

Understanding these common Rule-Based Techniques is crucial for cybersecurity professionals on the defensive side. By recognizing the patterns and behaviors that attackers exploit, defenders can implement more robust password policies and educate users on secure password practices.

In summary, Rule-Based Techniques are powerful tools in dictionary-based cryptanalysis. They take advantage of human tendencies and behaviors when creating passwords. Recognizing these techniques is essential for both attackers and defenders, as it informs strategies to crack passwords and protect against such attacks.

3.4 Strengths and Limitations [7, 4, 8, 5]

Rule-Based Techniques in dictionary-based cryptanalysis offer both strengths and limitations. Understanding these aspects is essential for both attackers seeking to maximize their success and defenders aiming to protect against such attacks.

3.4.1 Strengths

1. **Effectiveness Against Common Passwords** [7]: Rule-Based Techniques excel in cracking passwords that adhere to common patterns and predictable substitutions. Since many users opt for easily memorable passwords, attackers often find success by applying rules that target these tendencies. For example, passwords like "password123," "letmein," or "qwerty" are highly susceptible to rule-based attacks.

2. **Versatility in Generating Candidates** [7]: Rule-Based Techniques allow attackers to generate a wide range of candidate passwords efficiently. By applying various

rules, such as dictionary-based, keyboard patterns, and L33t speak rules, attackers expand their dictionary to encompass numerous possibilities.

3. **Customizability** [7]: Attackers can tailor rule sets to match specific targets or demographics. By analyzing target demographics or publicly available information, attackers can create rule sets that mimic the likely password choices of their victims.

4. **Rapid Cracking** [7]: For passwords adhering to common patterns, Rule-Based Techniques can lead to rapid cracking success. The predictability of human behavior in password creation simplifies the process for attackers.

3.4.2 Limitations

1. **Predictability** [7]: The very predictability that makes Rule-Based Techniques effective also serves as a limitation. Defenders can develop countermeasures to detect and block attacks employing well-known rules. Intrusion detection systems can flag multiple failed login attempts with rule-based patterns.

2. **Ineffectiveness for Complex Passwords** [7]: Rule-Based Techniques struggle when dealing with complex or unique passwords that do not conform to common patterns. Passwords containing a combination of unrelated words, random characters, or personalized acronyms are challenging for these techniques to crack.

3. **Password Strength Improvements** [7]: As organizations and individuals become more aware of cybersecurity threats, they often implement password policies that encourage the use of stronger passwords. Such policies may enforce minimum length requirements, character diversity, and the avoidance of common patterns. Rule-Based Techniques may be less effective against passwords created with these policies in mind.

4. **Limited for Targeted Attacks** [7]: While Rule-Based Techniques can be customized for specific targets, they may still fall short when dealing with individuals who have adopted robust security practices. Individuals who prioritize password security and employ techniques like passphrase creation or two-factor authentication pose greater challenges for rule-based attacks.

3.4.3 Data Suitability [7]

The effectiveness of Rule-Based Techniques also depends on the nature of the dataset being targeted. They are most potent when used against datasets where users tend to choose weak, easily guessable passwords. Such datasets may include:

- User accounts on websites with lax password policies. - Historical data with passwords created before the implementation of modern security practices. - Datasets acquired from breaches where hashed passwords are present.

In contrast, these techniques may be less successful against datasets from organizations with robust password policies or individuals who are security-conscious.

In summary, Rule-Based Techniques offer a potent approach to password cracking, particularly against commonly used and predictable passwords. However, their effectiveness is limited by the predictability of human behavior and the increased adoption of strong password policies. Understanding these strengths and limitations is crucial for both attackers and defenders in the field of cybersecurity.

3.5 Practical Applications [7, 4, 8, 5]

Understanding the practical applications of Rule-Based Techniques is essential in both offensive and defensive cybersecurity strategies. These techniques are widely used by attackers in attempts to compromise systems, and defenders must be well-versed in them to safeguard against dictionary attacks.

3.5.1 Offensive Use

Attackers leverage Rule-Based Techniques to launch dictionary attacks with the aim of cracking passwords and gaining unauthorized access to systems, accounts, or sensitive data. The practical applications of these techniques in offensive cybersecurity strategies include:

1. **Brute-Force Attacks** [7]: Attackers create dictionaries containing a wide range of candidate passwords generated through rule sets. These dictionaries may encompass dictionary-based rules, keyboard patterns, L33t speak variations, and more. By systematically testing these candidates against hashed passwords, attackers aim to identify successful matches.

2. **Password Guessing** [8]: Rule-Based Techniques enable attackers to guess passwords based on common patterns, known substitutions, and frequently used words. This approach is particularly effective when targeting users who choose easily guessable passwords, such as "password123" or "admin."

3. **Custom Dictionary Generation** [7]: Attackers can tailor their dictionaries to specific targets or demographics. By researching target information, such as names, interests, or affiliations, they can create dictionaries that mirror the likely password choices of their victims.

4. **Efficiency in Cracking** [5]: Rule-Based Techniques allow attackers to efficiently generate a large number of password candidates, increasing the chances of success in cracking passwords. This efficiency is particularly advantageous when dealing with datasets containing weak or commonly used passwords.

3.5.2 Defensive Use

Defenders must understand Rule-Based Techniques to strengthen cybersecurity measures and protect against dictionary-based attacks. Practical applications of these techniques in defensive cybersecurity strategies include:

1. **Password Policy Design** [4]: Organizations can use knowledge of common rule-based patterns to inform the design of password policies. Implementing policies that discourage easily guessable passwords and encourage complexity can enhance password security.

2. **Intrusion Detection** [7]: Intrusion detection systems (IDS) can be configured to detect patterns associated with rule-based attacks. Multiple failed login attempts with rule-based patterns can trigger alerts, allowing defenders to respond promptly to potential threats.

3. **User Education** [8]: Educating users about password security is crucial. By making users aware of common password pitfalls, such as using dictionary words or keyboard patterns, defenders can empower individuals to create more secure passwords.

4. **Password Strength Assessment** [4]: Organizations can assess the strength of user passwords by analyzing them for rule-based patterns. Passwords that exhibit such patterns may be flagged for review or require immediate change.

5. **Monitoring for Unusual Activity** [7]: Defenders can monitor systems for unusual or suspicious login activity, such as repeated login attempts with rule-based patterns. This proactive approach can help detect and mitigate attacks in real-time.

In summary, Rule-Based Techniques have practical applications in both offensive and defensive cybersecurity. Attackers use these techniques to compromise systems, while defenders employ them to bolster security measures and protect against dictionary-based attacks. Understanding the practical uses of these techniques is vital for cybersecurity professionals in their efforts to secure systems and data.

3.6 Conclusion [7, 4, 8, 5]

Rule-Based Techniques constitute a foundational pillar in the realm of dictionary-based cryptanalysis. These techniques, driven by their ability to generate passwords based on common human behaviors and tendencies, play a pivotal role in both offensive and defensive cybersecurity strategies. In this work, I embark on a journey to explore, expand, and enhance the landscape of Rule-Based Techniques.

My research seeks to push the boundaries of what Rule-Based Techniques can achieve. Specifically, I aim to develop novel algorithms for refining and customizing rule sets to maximize their effectiveness. By tailoring rules to the unique characteristics of targeted users or datasets, I endeavor to achieve superior results in password cracking and pattern recognition.

The significance of this research lies not only in its potential to advance the capabilities of dictionary-based cryptanalysis but also in its relevance to contemporary cybersecurity challenges. As data breaches and security threats continue to evolve, so must our methods of defense and analysis. Rule-Based Techniques, when optimized and adapted to specific contexts, can serve as powerful tools for both red teaming and blue teaming activities.

In the subsequent sections of this work, I will delve into the practical applications of Rule-Based Techniques, presenting experiments, and analyzing the results. Through these endeavors, I aim to contribute to the ongoing discourse on password security, intrusion detection, and the ever-evolving field of cybersecurity.

My exploration begins with an in-depth examination of the practical applications of Rule-Based Techniques in various cybersecurity scenarios. I will then detail the experiments conducted to assess the performance and efficacy of my customized rule sets. The results of these experiments will provide valuable insights into the strengths and limitations of my approach, shedding light on the path forward in the quest for enhanced password security.

4 Methods of Combining Rule-Based Techniques

4.1 Introduction

The amalgamation of Rule-Based Techniques is a crucial aspect of password security and dictionary-based cryptanalysis. These methods, when intelligently combined, contribute significantly to the diversity and complexity of generated password candidates. This section explores various techniques for combining rules, each offering unique advantages in the realm of offensive and defensive cybersecurity strategies.

4.2 Sequential Combination

4.2.1 Description

Sequential Combination involves a systematic chaining of different rules in a sequential order, resulting in a cascading effect where the output of one rule serves as the input for the next. This method allows for the creation of multi-step transformations, significantly increasing the complexity and diversity of generated passwords.

The process begins with the application of the first rule to a base password, generating an intermediate result. This intermediate result then undergoes further transformation as subsequent rules are sequentially applied. Each rule introduces a distinct modification, and the cumulative effect produces a final password candidate.

4.2.2 Example

Consider the application of a sequential combination in two steps: 1. Adding numbers to a base password: "secure" becomes "secure123." 2. Substituting specific characters, such as replacing 'e' with '3': "secure123" transforms into "s3cur3123."

This sequential combination results in a password with layered transformations, showcasing the potential complexity achieved through this method.

4.2.3 Advantages

- **Hierarchical Structure:** The sequential nature of this combination introduces a hierarchical structure, allowing for the creation of intricate and layered transformations.

- **Diverse Passwords:** By incorporating multi-step modifications, sequential combination enhances the diversity of generated passwords, making them more resilient against attacks.

This method draws inspiration from foundational works in cryptography, including [7, 4].

4.3 Parallel Combination

4.3.1 Description

Parallel Combination involves the simultaneous application of multiple rules to a base password, merging the results. This method enhances the variety of generated passwords.

The process includes the concurrent implementation of distinct rules, with each rule contributing to the transformation of the base password independently. The results of these parallel transformations are then combined to form a final password candidate.

4.3.2 Example

Consider the application of parallel combination with two simultaneous steps: 1. Adding numbers to a base password: "secure" becomes "secure123." 2. Applying L33t Speak transformations: "secure" transforms into "s3cur3."

The parallel combination results in a merged password candidate: "secure123s3cur3."

4.3.3 Advantages

- **Increased Diversity:** Simultaneously applying multiple transformations enhances the diversity of generated passwords, making them more robust against attacks.

- **Efficient Generation:** Parallel combination improves efficiency by generating a wide range of password candidates in a single step.

This method draws inspiration from foundational works in cryptography, including [6, 1].

4.4 Hierarchical Combination

4.4.1 Description

Hierarchical Combination organizes rules in a hierarchical structure, where certain rules may be applied to the results of other rules. This method allows for the creation of more sophisticated transformations.

In this approach, rules are structured hierarchically, forming a sequence where the output of one rule becomes the input for subsequent rules. This organization introduces a level of abstraction, enabling the development of complex transformations by combining multiple simpler rules.

4.4.2 Example

Consider the application of hierarchical combination in two steps: 1. Adding numbers to a base password: "secure" becomes "secure123." 2. Substituting letters with special characters: "secure123" transforms into "s3cur123."

In this example, the first rule (adding numbers) is a prerequisite for the second rule (substitution), illustrating the hierarchical relationship between them.

4.4.3 Advantages

- **Flexibility through Hierarchy:** Hierarchical organization provides flexibility by allowing rules to be applied in a structured order, enabling the creation of layered and sophisticated transformations.

- **Nuanced Patterns:** The hierarchical combination enables the development of nuanced and intricate patterns in generated passwords.

This method draws inspiration from foundational works in cryptography, including [7, 4].

4.5 Random Combination

4.5.1 Description

Random Combination involves the random selection of rules to apply, leading to the generation of diverse combinations. This method introduces an element of unpredictability.

In this method, at each step of password generation, rules are randomly selected for application. This introduces variability in the transformation process, making it challenging for attackers to predict patterns in the generated passwords.

4.5.2 Example

Consider the application of random combination in two steps: 1. Randomly choosing whether to apply a rule for substituting letters. 2. Randomly deciding whether to add numbers to the result.

This example showcases the variability introduced by the random selection of rules at each step.

4.5.3 Advantages

- **Variability for Enhanced Security:** The random selection of rules adds variability to the password generation process, making it more challenging for attackers to predict patterns.

- **Element of Surprise:** The element of unpredictability enhances the surprise factor in password generation, contributing to improved security.

This method draws inspiration from foundational works in cryptography, including [1, 2].

4.6 Selective Combination

4.6.1 Description

Selective Combination involves choosing specific rules based on predefined conditions or criteria. This method allows for targeted and controlled application of transformations.

In this method, rules are selectively applied based on predefined conditions. For example, a rule for substituting letters might be applied only to specific characters, such as

vowels. This selective approach provides control over which patterns are introduced into the password.

4.6.2 Example

Consider the application of selective combination: Applying a rule for substituting letters only to certain characters in a password, such as vowels.

This example demonstrates how the selective combination method can be employed to introduce specific transformations under predefined conditions.

4.6.3 Advantages

- **Targeted Application for Adaptability:** Selective combination allows for targeted application, enhancing the adaptability of password generation to specific criteria.

- **Control Over Introduced Patterns:** The method provides control over which patterns are introduced into the password, contributing to a more strategic approach.

This method draws inspiration from foundational works in cryptography, including [3, 6].

4.7 Iterative Combination

4.7.1 Description

Iterative Combination involves the repeated application of the same rules. This method allows for the reinforcement or stacking of specific transformations.

In this method, the same rule is applied multiple times to the evolving password. For instance, adding numbers to a base password and then applying the same rule again to the result.

4.7.2 Example

Consider the application of iterative combination: Adding numbers to a base password, then applying the same rule again to the result.

This example illustrates how iterative combination can be employed to intensify specific transformations, potentially increasing the chances of capturing desired passwords.

4.7.3 Advantages

- **Reinforcement of Specific Patterns:** Iterative combination reinforces specific patterns in the password, potentially increasing the chances of capturing desired passwords.

- **Creation of Intensified Transformations:** The method enables the creation of intensified transformations, contributing to the complexity of generated passwords.

This method draws inspiration from foundational works in cryptography, including [7, 4].

4.8 Composite Combination

4.8.1 Description

Composite Combination involves creating more complex rules by combining multiple simpler rules. This method allows for the development of intricate and multifaceted transformations.

In this approach, several basic rules are combined to form a composite rule, introducing a higher level of complexity to the password generation process.

4.8.2 Example

Consider the creation of a composite rule: Combining the rules of adding numbers, substituting letters, and other transformations into a single, more complex rule.

This example illustrates how composite combination can lead to the development of highly sophisticated and diverse patterns in generated passwords.

4.8.3 Advantages

- **Highly Sophisticated Patterns:** Composite combination enables the creation of highly sophisticated patterns in generated passwords by combining multiple simpler rules.

- **Broad Spectrum of Possibilities:** This method offers a broad spectrum of possibilities for password generation, contributing to increased diversity.

This method draws inspiration from foundational works in cryptography, including [7, 4, 6].

4.9 Conclusion

The methods of combining Rule-Based Techniques provide valuable tools for both attackers and defenders in the realm of cybersecurity. Understanding how to intelligently merge rules enhances the adaptability and effectiveness of password cracking strategies. By strategically combining patterns and rules, cybersecurity professionals can strengthen defenses, making it challenging for attackers to compromise systems.

5 Innovative Password Generation Algorithm

The primary focus of this research is to propose a novel password generation algorithm that combines Adaptive Hierarchical, Parallel, Selective, Iterative, and Composite Combination techniques. My goal is to create a method that surpasses existing rule-based combination approaches by leveraging the strengths of each strategy. The proposed algorithm aims to provide superior results in terms of password diversity, strength, and resistance to various attacks.

5.1 Motivation

The motivation behind this research stems from the need for more advanced and adaptable password generation methods. Existing techniques often face challenges related to predictability, limited diversity, and susceptibility to modern attack strategies. By combining multiple strategies, I aim to address these challenges and contribute to the development of robust password generation techniques.

5.2 Methodology

The proposed algorithm integrates the following techniques:

- **Adaptive Hierarchical Combination:** Utilizing a hierarchical tree structure where rules are dynamically adjusted based on their historical performance. This adaptability ensures effective rules are prioritized, contributing to the algorithm's ability to adapt to varying password complexities.
- **Parallel Combination:** Simultaneously applying multiple rules at each level of the tree, introducing diversity by creating variations in password candidates concurrently.
- **Selective Combination:** Incorporating a selective mechanism to intelligently choose rules based on predefined conditions. This selective approach provides controlled and targeted application of transformations, enhancing adaptability.
- **Iterative Combination:** Applying some rules iteratively to reinforce specific patterns in the password. This iterative process contributes to the creation of intensified transformations.
- **Composite Combination:** Creating complex rules by combining simpler rules, expanding the algorithm's exploration of diverse password patterns.

5.3 Algorithm Execution

The proposed algorithm executes the following steps:

1. **Initialization:** Begin with the initialization of a base password and the creation of an initial tree structure.

2. **Adaptive Hierarchical Adjustment:** Dynamically adjust rules at each level of the tree based on their historical performance.

3. **Parallel Transformation:** Simultaneously apply multiple rules at each level to transform the base password concurrently.

4. **Selective Application:** Selectively apply rules based on specific conditions or criteria.

5. **Iterative Reinforcement:** Iteratively apply some rules to reinforce specific patterns in the password.

6. **Composite Rule Generation:** Generate complex rules by combining simpler rules.

7. **Tree Expansion:** Expand the tree structure at each level to generate a broader set of potential password candidates.

8. **Password Candidate Selection:** Select top-performing passwords from the generated candidates based on predefined criteria.

5.4 Evaluation and Validation

The proposed algorithm's effectiveness will be evaluated through extensive comparisons with existing rule-based combination techniques. Evaluation criteria will include password diversity, strength, and resistance to various attack scenarios. Established password datasets and benchmarking metrics will be utilized to validate the algorithm's efficiency.

5.5 Conclusion

The proposed Adaptive Hierarchical-Parallel-Selective-Iterative-Composite Combination Algorithm aims to offer a versatile and robust approach to password generation. By combining multiple strategies and adapting dynamically, I anticipate that my algorithm will outperform existing methods in terms of password diversity and strength. The subsequent chapter will present the experimental setup, results, and a comprehensive analysis of the proposed method's performance.

6 Comparison Methodology

To assess the effectiveness of the proposed Adaptive Hierarchical-Parallel-Selective-Iterative-Composite Combination Algorithm, a comprehensive comparison will be conducted against existing rule-based combination techniques. The evaluation will focus on the generation of passwords derived from a given word (e.g., "example"). The goal is to explore how well the algorithm can generate complex passwords based on common user practices.

6.1 Password Derivation from Base Word

The first aspect of the comparison involves generating passwords based on a base word using various rules. The base word will be manipulated through rule-based transformations to create a set of potential passwords. The effectiveness will be measured in terms of diversity, complexity, and resistance to common attack strategies.

6.2 Personalization Mechanism

Additionally, the proposed algorithm includes a personalization mechanism. Users can input specific information about an individual, and the algorithm will adapt its rule application to generate passwords tailored to that person. This personalization feature will be compared against generic rule applications to evaluate its impact on password strength.

6.3 Performance Metrics

The evaluation will consider the following performance metrics:

- **Password Diversity:** The variety of passwords generated, measured by the number of unique patterns.
- **Password Complexity:** The strength and complexity of passwords, assessed through metrics like entropy and resistance to pattern analysis.
- **Resistance to Attacks:** Testing the generated passwords against common attack scenarios, including brute-force attacks, dictionary attacks, and rule-based attacks.

6.4 Experimental Setup

The experiments will utilize established password datasets and benchmarking metrics. The comparison will involve running the proposed algorithm and existing rule-based methods on the same datasets under controlled conditions.

6.5 Result Analysis

The results will be analyzed to determine the algorithm's effectiveness in password generation. Insights gained from the comparison will inform the algorithm's strengths, weaknesses, and areas for potential improvement.

This methodology aims to provide a comprehensive understanding of how the proposed algorithm performs in generating secure and diverse passwords compared to existing rule-based techniques.

7 Implementation of the Innovative Password Generation Algorithm

In this section, I delve into the details of implementing the innovative password generation algorithm. I provide insights into key aspects, including the algorithm's structure, employed techniques, and the process of generating and optimizing passwords. Practical examples will be presented to illustrate how the approach differs from existing methods.

7.1 Algorithm Structure

The password generation algorithm is designed with a focus on combining Adaptive Hierarchical, Parallel, Selective, Iterative, and Composite Combination techniques. This section outlines the hierarchical tree structure, parallel rule application, selective mechanisms, iterative processes, and composite rule generation.

7.2 Password Generation Process

I walk through the step-by-step execution of the algorithm, covering the initialization of a base password, adaptive hierarchical adjustments, parallel transformations, selective rule applications, iterative reinforcements, composite rule generation, tree expansion, and the selection of top-performing password candidates.

7.3 Practical Examples

To enhance understanding, practical examples demonstrate the application of the algorithm to generate diverse and complex passwords. These examples showcase the adaptability and strength of the approach in creating passwords resistant to various attack scenarios.

7.4 Optimization Techniques

Efforts to optimize the algorithm's performance will be discussed, addressing considerations such as computational efficiency, scalability, and the trade-offs involved in balancing password diversity and generation speed.

7.5 Result Analysis

The analysis focuses on the results obtained from applying the algorithm to evaluate its effectiveness in terms of password diversity, strength, and resistance to common attack scenarios. Insights gained from this analysis will inform potential refinements and improvements.

This section offers a comprehensive overview of the implementation details of the innovative password generation algorithm, providing readers with a deeper understanding of its inner workings and practical applications.

8 Literature Review

References

[1] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. Springer, 2004.

[2] Jeremy M. Carroll and Paul C. van Oorschot. Predicting password guessability for an account creation task. *Proceedings of the 17th ACM conference on Computer and Communications Security*, 2010.

[3] S. Komanduri, R. Shay, P. G. Kelley, M. L. Mazurek, L. Bauer, N. Christin, L. F. Cranor, and J. Hong. Of passwords and people: Measuring the effect of password-composition policies. *CHI 2011 Proceedings*, 2011.

[4] Christof Paar and Jan Pelzl. *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer, 2015.

[5] Dario Pasquini, Marco Cianfriglia, Giuseppe Ateniese, and Massimo Bernaschi. Reducing bias in modeling real-world password strength via deep learning and dynamic dictionaries. *Journal of Cybersecurity*.

[6] Bruce Schneier. *Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World*. W. W. Norton & Company, 2015.

[7] William Stallings. *Cryptography and Network Security*. Pearson, 2020.

[8] Fangyi Yu. On deep learning in password guessing: A survey. *International Journal of Information Security*.

9

Temat pracy: "Optymalne generowanie słowników dla potrzeb kryptoanalizy słownikowej"
Opis pracy:

Kryptoanaliza słownikowa ukierunkowana jest na odkrycie haseł (tajnych fraz tekstowych) przechowywanych w systemach komputerowych w postaci tzw. hashy. czyli wiadomości wytwarzanych z frazy hasłowej użytkownika przy pomocy funkcji skrótu lub funkcji KDF (Key Derivations Function). Najważniejsza cecha tych funkcji jest jednokierunkowość, a wiec z hasha nie można odzyskać hasła. Atak słownikowy jest odmiana ataku siłowego (czyli systematycznego sprawdzania wszystkich możliwych haseł) i polega na przygotowaniu dużego słownika zawierajacego potencjalne hasła i nastepnie testowaniu, czy któryś z tych haseł wytworzy atakowany hash. Trafność doboru słownika ma olbrzymie znaczenie dla efektywności takiego ataku.

Ataki tego typu sa bardzo popularne w ostatnim czasie, gdy czesto dochodzi do wycieku baz danych zawierajacych dane użytkowników (loginy, emaile, telefony i właśnie hashe haseł). Z drugiej strony organy ścigania czesto spotykaja z koniecznością odzyskania haseł z materiału dowodowego. W tym drugim przypadku pożadana jest optymalizacja polegajaca na zredukowaniu listy potencjalnych haseł do takich, które sa najprawdopodobniej używane przez podejrzanego. Zakładamy, że zgromadzony materiał dowodowy może dostarczyć wskazówek co do najbardziej prawdopodobnych haseł. Celem pracy jest zaproponowanie i przetestowanie metody efektywnego generowania słowników na podstawie danych pozyskanych z materiału dowodowego. Preferowanym rozwiazaniem jest wykorzystaniem nowych mechanizmów do generowania haseł na podstawie znanych reguł, w celu uzyskania innego zbioru prawdopodobnych haseł z wykorzystaniem tych samych reguł z których korzystaja dostepne algorytmy.