A New Multimodal Approach for Password Strength Estimation—Part I: Theory and Algorithms

Javier Galbally, Iwen Coisel, and Ignacio Sanchez

Abstract—**After more than two decades of research in the field of password strength estimation, one clear conclusion may be drawn: no password strength metric by itself is better than all other metrics for every possible password. Building upon this certainty and also taking advantage of the knowledge gained in the area of information fusion, in this paper, we propose a novel multimodal strength metric that combines several imperfect individual metrics to benefit from their strong points in order to overcome many of their weaknesses. The final multimodal metric comprises different modules based both on heuristics and statistics, which, after their fusion, succeed to provide in real time a realistic and reliable feedback regarding the "guessability" of passwords. The validation protocol and the test results are presented and discussed in a companion paper.**

Index Terms—**Password security, strength metrics, information fusion, multimodality, Markov chains, password policies, privacy.**

I. Introduction

How secure are passwords? Such a question has recently been brought once again to the public fore following the release in May 2016 of over 117 million pairs of password hashes and emails of LinkedIn acocunts [1]. Just two days after the leak, KoreLogic announced that they had been able to crack over 65% of the hashes in the first two hours of a password guessing session [2]. As an illustrative anecdote, it was later known that one of the affected LinkedIn users was one of the founders and current CEO of Facebook. His password was not only among the cracked ones but had also been reused in other online services, which enabled hackers to also access both his Twitter and Pinterest accounts [3].

The answer to the question, how secure are passwords? is ambiguous: it depends on the password. Certain passwords are more secure, or stronger, than others. As such, password strength is commonly understood as a way to measure how difficult it is to break passwords. Therefore, password strength metrics are functions that take as input a password and output a score related to the strength of that password.

The purpose of password strength metrics is the identification of weak passwords in order to support password policies that attempt to maximize the security offered by this type of user authentication [4]. A common way to enforce the policy is the real-time checking of the password strength during the password selection process by the users. This good practice was already promoted by researchers in the field of password security in the early 1990's [5]: should the strength fall below an acceptable threshold, the user will be requested to provide a different password. The effectiveness of a password strength metric in estimating the actual resistance of passwords will have a direct impact on the level of security for both the users and the provider [6].

Coming back to the recent LinkedIn leak scandal, the impact could have been reduced if users had not been allowed to select weak passwords, or at least warned of the danger to do so, specially if the password is later reused is later reused [7].

Current password strength estimation methods may be classified into three main groups, each of them based on a different perspective of the same problem: attack-based, heuristic-based and statistical-based (further details about the rationale behind these methods are given in Sect. III).

- **Attack-based methods**. Such methods give a measure of the resistance of passwords depending on the time taken by a specific attack (or set of attacks) to break it [8]–[10] (see Sect. II for further details on existing password attacks). The longer it takes for the attack to break the password, the stronger it is.
- **Heuristic-based methods**. These methods focus on providing a measure of the password complexity based on heuristics [11], [12]. The *de-facto* standard for this type of methods is the NIST 800-63 published in 2004 and updated in 2012 [12]. It proposes to measure password strength in entropy bits, following Shannon's Information Theory [13], on the basis of some simple rules such as the length of the password and the type of characters used (e.g., low-case, upper-case, or digits).
- **Probabilistic-based methods**. Searchingto address the shortcomings of the previous techniques, new methods for password strength estimation have emerged based on the statistical evaluation of passwords [14], [15]. Most of these methods are based on Markov Models [16] (please see Sect. VI for further details on Markov Models).

A key factor that has been traditionally overlooked by many of the password strength estimation approaches mentioned above is that: password strength is not a universal value. Rather, it is highly dependent on the context in which the password is used [17]. That is, the exact same password

Manuscript received May 23, 2016; revised September 17, 2016 and November 22, 2016; accepted November 22, 2016. Date of publication December 7, 2016; date of current version August 29, 2017. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Karen Renaud.

The authors are with the Joint Research Center, European Commission, 21027 Ispra, Italy (e-mail: javier.galbally@jrc.ec.europa.eu; iwen.coisel@jrc.ec.europa.eu; ignacio.sanchez@jrc.ec.europa.eu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2016.2636092

can have a significantly different strength depending on the application where it is utilized. For example, a Hungarian word used to login to a computer in a Spanish-based company can be a quite strong password. However, that same password, used in a Hungarian-based context would most likely be regarded as weak. A similar argument can be made for a five character password used to access remotely an account protected by a three-attempt limit (strong), compared to that same password used to access a local service with unlimited number of attempts (weak).

The current scene in password strength estimation presented above leads to two main conclusions:

- **Conclusion 1: No golden bullet**. As happens for most complex problems, in password strength estimation there is no unique valid solution. That is, no password strength metric by itself is better than all other metrics for every possible password. All existing strategies for password strength estimation present advantages and drawbacks (these are presented in Sect. III). Therefore, new methods could potentially benefit from the strong points of each of the three types of approaches mentioned above in order to overcome their weaknesses.
- **Conclusion 2: No immutability**. Password strength estimation algorithms should not be immutable. On the contrary, they should be able to adapt to different environments in order to give more accurate strength values (e.g., depending on the language, alphabet, etc.)

These conclusions have led to set two main objectives in the design and development of the new password strength estimation method presented in the article, which can hopefully contribute to the advance of the state of the art in password strength estimation:

- **Objective 1: Multimodality**. The first main by-design goal of the novel scheme is to exploit the advantages of different methods in order to provide a more reliable feedback regarding the robustness of passwords than each of the algorithms individually. To do so, the model combines the strength values generated by different complementary modules based on: 1) specific attacks, 2) heuristics and 3) statistics. Such multimodality approach, as will be explained in Sect. IV, is supported by multiple works in the field of information fusion.
- **Objective 2: Adaptability**. The second main objective of the present work is to overcome the typical shortcoming of current methods that provide one unique strength value for each password independently of the context where it is used. This is achieved by devising a strength estimation algorithm that may be adjusted, through a training process, to diverse application-specific environments.

With these two major objectives in mind, the contributions of the present Part I of this series of two papers may be summarized as follows (for the experimental evaluation of the method we refer the reader to Part II [18]): 1) the global fusion strategy to provide a unique strength indicator through the combination of several individual modules (see Sect. IV); 2) the two individual probabilistic-based strength metrics described in the paper are novel: the Markov Chain with adaptive memory (see Sect. VI-A) and the Hierarchical Markov Chain (see Sect. VI-B).

The rest of the article is structured as follows. In order to better set the scene that has promoted the development of the new multimodal strength metric, two brief introductions to password guessing attacks and to password strength metrics are given in Sects. II and III, respectively. The overall multimodal metric is described in Sect. IV, with each of the individual modules presented in Sects. V, VI and VII. Sect. VIII specifies the contributions of the proposed approach and discusses similarities and differences with previous methods. Finally, conclusions are given in Sect. IX.

II. Introduction to Password Guessing Attacks

As mentioned in the introduction, the main objective of a strength metric is to estimate the "guessability" of a password (understood as the ease with which a password can be guessed). This parameter is undoubtedly linked, among other aspects, to the type of guessing attacks that will be performed to break it. For this reason, the present section provides some basic concepts related to the field of password guessing that can help the reader to better understand the rationale behind the multimodal strength metric proposed in the work.

Why are password guessing attacks successful? Why are they able to consistently break a significant amount of passwords? Since the development of the first password cracking algorithms [20], the answer to these questions has been relatively simple: passwords are chosen by humans, and humans are, to some extent, predictable [21], [22]. In fact, humans have a tough time when they are asked to be random, and even a tougher time when they are asked to "remember random" [23], [24]. This way, in many occasions, users deliberately select passwords easy to remember and, as a result, easy to guess [25]. Password guessing attacks take advantage of this predictability to give preference and test first those passwords that are more likely to be selected by a person in order to speed up the guessing process (see Fig. 1).

With this objective in mind, as shown in Fig. 1, adversaries usually apply a sequential strategy in their password guessing sessions: starting by the most straightforward attacks (i.e., internal squares in Fig. 1) and gradually moving towards the more general ones (i.e., outer squares in Fig. 1). This means that the first attacks to be carried out are those that have traditionally shown a higher success rate measured in terms of passwords cracked per given number of attempts. These initial attacks are very fast retrieving passwords at the beginning but also become unsuccessful soon, as they explore a limited region of the whole password space. The last attacks to be implemented are the most general ones, that is, those that, in absolute terms, are capable of cracking the largest amount of passwords but that, in turn, also generate many more wrong guesses which makes them slower (i.e., they cover a large region of the whole password space). Such a password guessing strategy has even been implemented in automatic tools that sequentially launch the most efficient attack as the number of guesses increases [26], [27]. Just as an illustrative

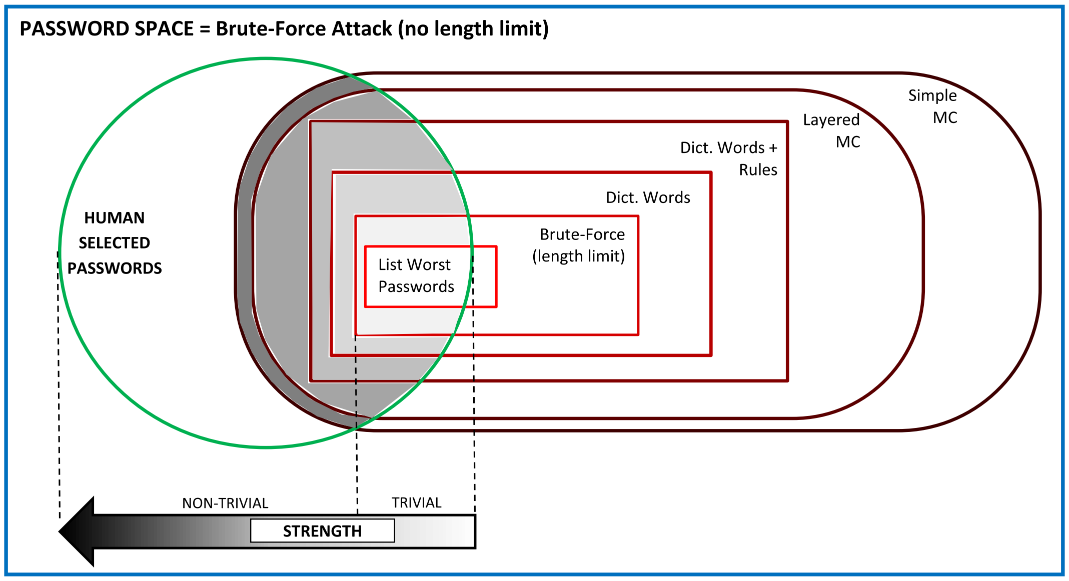

Fig. 1. General diagram of the password-cracking problem. The external rectangle represents the space of all possible passwords. The circle represents the limited subspace of human-selected passwords. The internal rectangles represent the subspaces of computer-generated passwords for different attacks: from the most targeted ones (i.e., inner rectangles that cover a smaller subset of the whole space) to the most general ones (i.e., outer rectangles covering wider subspaces). The attacks that appear are just a typical subset example selected for illustrative purposes. Regular rectangles represent attacks based on heuristics, while rounded-corner rectangles represent probabilistic-based attacks (the Layered Markov Chain and the Simple Markov Chain are described in [19]). The shaded areas (overlap between human-selected passwords and computer-generated passwords) represent the cracked passwords. A darker shade corresponds to passwords that, on average, are harder to crack (i.e., it takes a higher number of guesses to retrieve them). The bottom arrow represents password strength, where a darker shade corresponds to passwords that are more difficult to guess (i.e., stronger).

example, a typical password guessing session would follow the next two-step methodology [28]–[30]:

- **STEP 1: Break trivial passwords**. Trivial passwords may be defined as those that can be broken, with all certainty, within a limited number of guesses. Therefore, these passwords should receive very low strength values. In general, they are passwords vulnerable to:
  - *Attack 1: Attack based on a list of known popular passwords* [31]–[33]. It simply attempts to gain access trying all the possibilities in any of the existing lists that contain well-known common passwords [34]–[36], e.g., “123456”, “admin123”, “password”. These lists, in some cases released on a yearly basis [37], are compiled according to the number of times a given password appears in the existing large databases of passwords [38].
  - *Attack 2: Brute-force attack* [39], [40]. This attack performs an exhaustive search over all possible combinations within a given search space defined by an alphabet of N symbols (e.g., $N = 26$ lower-case letters in the ASCII code) and a given password length L. The total number of guesses is computed as N^L. If the size of the search space is too big, the attack becomes practically unfeasible.
- **STEP 2: Break non-trivial passwords**. Other increasingly complex guessing algorithms are applied for passwords that are robust to the previous two attacks. These passwords are non-trivial in the sense that there is only a certain non-zero probability (i.e., no guarantee) that they will be retrieved. As such, they should be assigned a higher strength by password strength metrics.
  - *Attack 3: Dictionary attack* [39]. It is similar to the attack based on a list of very common passwords but in this case a dictionary (i.e., a very large list of words) is used instead [43]. In fact, in many cases, these dictionaries already integrate lists of known passwords. The basis of this approach is that in most cases humans select real words as passwords since they are easier to remember [44]. Different cracking tools specialized in performing this type of attacks may be found in the internet [27], [45], [46].
  - *Attack 4: Dictionary attack with rules* [20]. Analogue to the dictionary attack but in this case not only the words in the dictionary are used, but also slight modifications of them, that are produced following certain rules commonly observed in the generation of passwords such as: capitalize the first letter, add numbers at the end, etc.
  - *Attack 5: Probabilistic attacks* [19], [44], [47], [48]. Although dictionary attacks are really effective, being able to retrieve up to 50% of the passwords of all known big datasets, they are not able to produce passwords which are not directly derived following a fixed set of rules applied to a selected dictionary. As an alternative to break passwords resistant to dictionary attacks, probabilistic approaches apply statistical models, in most cases based on some variation of Markov Models [49], that try to capture the way in which humans produce passwords. Such models are sampled to generate human-like guesses.

In the previous typical example, the first two simple but highly efficient attacks designed to break trivial passwords may be considered substantially as a *de-facto* standard. These two attacks will be applied, with very small variations, at the beginning of virtually any password guessing session.

On the other hand, attacks 3-5 applied to break non-trivial passwords present uncountable variations. For example, dictionary-based attacks depend on the dictionary considered and on the type of rules implemented. Probabilistic attacks depend on the statistical model selected and also on the data used to train that model. As a result, it is not feasible to foresee all the precise attacks that will be used against non-trivial passwords, specially because new guessing algorithms are being continuously devised.

The reader should bear in mind that the previous typical password guessing session and the subset of attacks considered in it, are just given for illustrative purposes and are not exhaustive (specially those attacks targeting non-trivial passwords). It should also be highlighted that the frontier between trivial and non-trivial passwords, although quite well defined, is not deterministic. As will be explained in Sect. V, such categorization is context-dependent, since not the same passwords are brute-forceable in all applications.

III. Introduction to Password Strength Metrics

All passwords can be broken. Given enough time, eventually, any password will be broken by means of an unlimited brute-force attack (as shown by the external rectangle in Fig. 1). Therefore, the question is not any more *if* a password can be guessed, but rather, *when* it will be guessed. In fact, as mentioned in the previous section, all password guessing attacks have one common aim: accelerate the speed at which passwords are retrieved. As such, it seems reasonable to think of *time* as the ultimate scale to measure the strength of passwords. A password which is broken in 10 minutes is weaker than one which is broken in 10 days, 10 months, 10 years or 10 centuries.

However, time, as a metric for password strength, is very dependent on external variables, for instance, the time needed to crack a password using a certain attack can be drastically reduced if the computational resources are increased. But not only computational power affects the password-guessing time, it is also intimately related to the type of oracle that discloses the validity or not of the password. As an example, if the oracle allows an offline attack (i.e., the attacker is in possession of the hash of a password), hundreds of millions of candidate passwords can be tested per second for some currently wide-spread used algorithms using an optimized implementation for GPUs [50]. On the other side of the spectrum, if the oracle is a remotely accessible authentication system, there can be a limit on the allowed amount of passwords to be tested and eventually the account will be temporarily or permanently locked out [51]. Furthermore, the type of hashing or encryption algorithm also has a very important impact on the time needed to perform each comparison (e.g., attacking MD5 hashes is significantly faster than attacking SHA-3 ones.[1])

[1]Our own experiments carried out running a benchmark of Hashcat 3.10 with two AMD Radeon R9 290 cards showed the next results: MD5 cracking speed was $20 \cdot 10^9$ Hash/sec, while SHA-3 cracking speed was $0.3 \cdot 10^9$ Hash/sec, which is in line with values reported in [50].

For the above mentioned reasons, time is usually converted into "number of guesses required to break a password" (NoG), which is a strength unit independent of the actual contextual conditions. Then, depending on the "NoG per second" that can be performed in a given framework, it is straightforward to estimate the time that will take to break a certain password. Some initiatives, such as the NIST recommendation [12], go a step further and translate the "number of guesses" into "number of bits" as a way to measure the password strength in terms of a fictional password entropy [13]. For instance, a password that is cracked by an attack in 1024 guesses would mean that has a 10 bit strength entropy. As such, all three measures, *time*, *number of guesses* and *entropy* (*or number of bits*), are three directly related ways of measuring the same concept. In the following, we will refer to NoG as the standard scale to measure password strength.

As mentioned in Sect. I, from a general perspective, password strength estimation approaches can be classified in three big groups: attack-based, heuristic-based and probabilistic-based. The next subsections discuss the rational behind each of the groups, their advantages and limitations. The reader should be aware that Sect. VIII further analyzes some of the methods mentioned here and compares them to the multimodal strength metric presented in this work, highlighting similarities and differences.

A. Attack-Based Approaches

Probably, one of the most direct ways to objectively quantify NoG is to actually attack a password until it is broken. Most of the works in the state of the art that consider this type of approach are based on a single-attack strategy [8]–[10], [52], [53]. However, a recent work has shown that simulating a cracking-session with multiple successive attacks of increasing complexity provides significantly better results [30].

These attack-based approaches present two major issues: 1) on the one hand, they are not able to generalize well to different attacks or even to different configurations of the same attack [30] (e.g., using different dictionaries). That is, assuming that it takes 1,000 guesses to break a password with one given attack, we have a measure of its strength against that particular attack, but, what if the attack is changed? How can the NoG associated to a given password be made independent of the attacks considered? 2) On the other hand, carrying out an attack to compute the NoG needed to break a password requires time, and, for many applications, an immediate answer is needed (e.g., giving feedback to a user regarding the strength of a password he has just chosen).

Due to the two limitations specified above, the objective and real-time evaluation of NoG is difficult and, although some very interesting strength checkers have been designed following this approach [9], alternative ways of indirectly *estimating* password strength have been proposed.

Nevertheless, attack-based approaches are extremely helpful to detect weak passwords vulnerable to very targeted attacks that do not admit multiple configurations, such as the ones using lists of common trivial passwords (see Attack 1 in Sect. II). This has lead to different publications that recommend the use of black-lists of banned passwords as part of the protection methods [31], [54].

B. Heuristic-Based Approaches

Currently, probably the most popular and deployed trend in the field of password strength metrics is that based on heuristics also known as *LUDS*: counts of lower- and upper-case letters, digits and symbols. Although some very valuable heuristic-based works are currently moving away from the LUDS paradigm [55], the majority of password policies used in practice are based on this concept [54]. This strategy is followed, for instance, by the current NIST recommendation [12], by different password strength estimators that may be found on-line [56], and even by top technological companies [57]. Such metrics are based on expert rules derived in an ad-hoc manner from experience, that aim at estimating the complexity of passwords under the assumption that a more complex password (according to these principles) will be less vulnerable to attacks. Examples of these rules are: longer passwords are more robust, or passwords with more character types (e.g., lower case, upper case, numbers and special characters) are more robust.

Although such rules, from a general point of view, are sensible, they may produce illogical results for many particular passwords. In fact, it has now been shown in different works that these methods are not reliable strength estimators for a wide range of passwords [8], [10], [31] and have even been the target of witty critics outside the security community [58]. For instance, these strength metrics will assign a higher strength to a password like "David-1982" than to "RpixTsGa". "David-1982" is length 10 and has lower case, upper case, numbers and special characters, whereas "RpixTsGa" is length 8 and has only lower case and upper case letters. However, most users would agree that "David-1982" is a much more guessable password than "RpixTsGa", that is, it would take a higher number of guesses to break "RpixTsGa" than "David-1982". As a consequence of the comments received, NIST is currently reviewing its recommendation and has released a new draft that clearly changes the philosophy of previous versions towards new more comprehensive guidelines not only focused on rules but, for instance, recommending the use of blacklists built based on available password corpuses [59].

The question is, why do heuristic-based metrics fail to give, in many cases, a correct estimation for the strength of passwords? The problem lies in the fact that, implicitly, these metrics rely on the incorrect premise that: all passwords are equally likely to be selected by humans. That is, the probability of occurrence of a password like "kzlprtu" would be the same as that of "rebecca". Under this assumption, the usual proposed ad-hoc rules hold: longer passwords are more robust, and passwords that contain more character types are more robust. In fact, in this unrealistic case, the only sensible approach for breaking passwords would be to use brute-force, as no other attack would outperform it in terms of number of guesses. However, as mentioned in Sect. II, experience has shown that human behaviour for password selection is significantly different from the fictional situation presented here in which all passwords were equally probable. Rather, it is now known that human password selection is heavily influenced by natural language, which leads to certain passwords being much more likely to be chosen by users than others [44].

The discussion given above leaves a very important lesson: heuristic-based methods, in spite of their limitations, should in no case be dismissed, as they are the optimal way of detecting passwords vulnerable to brute-force attacks (see Attack 2 in Sect II). As such, a comprehensive strength metric should, in any case, take heuristics into account.

C. Probabilistic-Based Approaches

The previous subsection on heuristic-based methods introduced a very interesting concept: the relationship between passwords and probabilities. It seems reasonable to use the probability that a password is selected by a human as an estimation of the NoG it will take to break it, since also "smart" attacks exploit the lack of randomness in the selection of passwords. This way, the probability of occurrence may be used as an effective password strength metric. If such probability is very high, the strength is low, and the other way around, a low probability would denote high strength. However, the problem is not simple: The probability that a password is selected by a human is intrinsically a very subjective matter. Therefore, the ultimate question is, how can it be estimated?

The most straight forward manner to answer the previous question would be to count the number of times that a given password appears in currently available databases. This would lead to what has been referred to in some works as the "ideal password strength meter" [17]. However, except for very few examples (such as "password", "12345" or "test123" that are the basis for Attack 1 in Sect. II), most passwords do not appear enough times in order to reliably estimate their probability of occurrence. Furthermore, many likely passwords are not found in public databases and therefore would be assigned an unrealistic probability of occurrence equal to zero.

In order to address this issue, a new branch of strength metrics based on statistical models was introduced. Similarly to probabilistic attacks, statistical strength estimation methods try to accurately model the likelihood that a password is selected by a user. In fact, many of these algorithms can be indistinctively used, with very small variations, to produce password guesses or to estimate the strength of passwords. The main challenge to be faced in the development of these methods is: given a finite set of training data containing human selected passwords (ideally as large as possible), generate a statistical model which is able to accurately represent the probability of occurrence of all possible passwords and not just those contained in the training set. Therefore, in general, this type of methods require to reach a difficult compromise between generality (i.e., ability to correctly estimate the strength of new unseen passwords) and accuracy.

Since the first cracking methods using probabilities appeared to reduce the size of dictionaries in dictionary-based attacks [60], many different probabilistic strength estimation algorithms have been proposed. The vast majority of these techniques take advantage of the lessons learned in the field

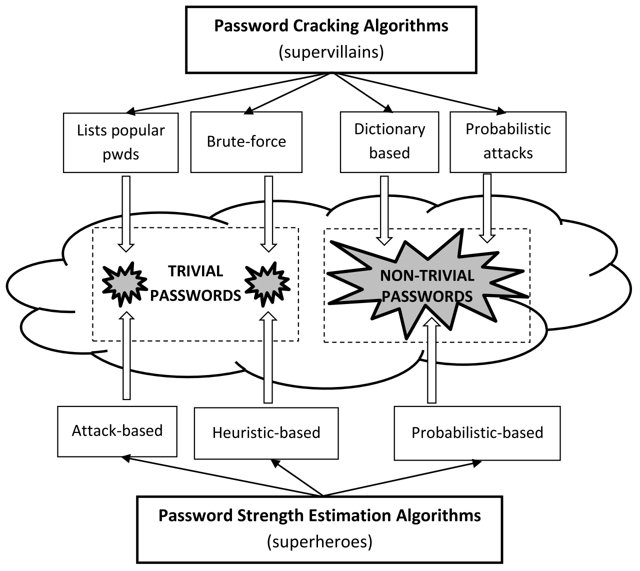

Fig. 2. General diagram of the "password-cracking *vs* strength-estimation" problem.

of natural language modeling [68] and may be grouped in three main trends: 1) Approaches that model passwords as simple character sequences using some form of the popular Markov Models [15], [17], [19], [47], [62]. 2) A very successful line of research using Probabilistic Context-Free Grammars (PCFG) was initiated by Weir as a cracking tool [48] that was further refined in [63]. The method was later adapted as a strength metric [64]. The core idea behind this innovative approach is to consider not only characters, but also the existence of higher-order structures or semantic patterns. 3) Lastly, Bonneau addressed the problem of determining the probability of occurrence of passwords from a more theoretical perspective using different deterministic mathematical functions (instead of HMM or PCFG) to approximate the statistical distribution of passwords to finally propose his α-guesswork metric [10].

While probabilistic-based approaches have typically outperformed heuristic-based or attack-based methods for the robustness estimation of non-trivial passwords, they usually fail to assign sufficiently low strength values to trivial passwords (as defined in Sect. II).

D. Current Password Strength Metrics: Summary

Given the argumentation presented in this section, it follows that, however important and useful password strength estimation is, it remains a very challenging problem with no closed answer or unified approach to address it. As already introduced in Sect. I and further discussed here, all methods have advantages and drawbacks.

The above statement about pros and cons of current strength estimation approaches implies that: no method is universally better than all the rest for every password, but also, no password strength estimation method is fully useless or discardable. In fact, each of the three type of approaches described in the present section are very effective at accurately producing strength results for different specific sets of passwords:

- Attack-based approaches using blacklists are optimal to detect trivial passwords vulnerable to attacks based on lists containing the most trivial passwords (see Attack 1 in Sect. II).
- Heuristic-based approaches are optimal to detect trivial passwords vulnerable to brute-force attacks (see Attack 2 in Sect. II).
- Probabilistic-based approaches are optimal to detect non-trivial passwords which may be vulnerable to: dictionary attacks or probabilistic attacks (see Attacks 3-5 in Sect. II).

As such, in this on-going battle between good (i.e., password strength metrics) and evil (i.e., password cracking algorithms) where passwords are the battlefield, it happens that each supervillain has its own nemesis, as depicted in Fig. 2. However, no superhero can fight on its own all supervillains: An alliance needs to be made.

Such a comic-ish analogy leads to the core idea behind the proposed multimodal strength metric described in the following sections: do not dismiss certain imperfect password strength approaches leaving all the responsibility of correctly estimating the robustness of passwords to just one algorithm. Instead, combine different complementary methods specialized in detecting a particular group of weak passwords to generate a more general, accurate and reliable overall approach. In simple terms, follow the principle "strength in numbers" to try to defeat the enemy, as depicted in Fig. 3.

IV. Multimodal Strength Metric

In the present section we build upon the general analysis provided previously in Sects. II and III to propose a novel method that meets by design the two main goals highlighted in the article introduction: **multimodality** and **adaptability**.

The main rationale behind the proposed approach for password strength estimation is that: by exploiting the capabilities of different individual techniques through their fusion, it will be possible to achieve one unique *multimodal* measure which overcomes many of their weaknesses. Such a "multimodal hypothesis" is supported by different works carried out in the active area of knowledge of information fusion that has been applied to multiple real-world problems.

In particular, the present strength metric is inspired by information fusion systems that exploit multimodality to improve the overall performance of individual solutions for challenging problems such as biometrics [65]. These systems consolidate the evidence presented by several sources and typically provide better performance compared to systems based on a single source. Although information fusion in a typical multimodal machine-learning system can be performed at various levels, integration at the score level is the most common approach. In the present work we take advantage of the knowledge gained in the field of score fusion in machine-learning systems, and apply it to generate a novel and more robust password strength metric (please see Sect. VII for further details on how the fusion is performed).

As can be seen in Fig. 4, according to the multimodality principle presented above, the proposed approach takes as input a password and generates one single strength score S_{MULTI}. To do this, inspired by the typical two step guessing

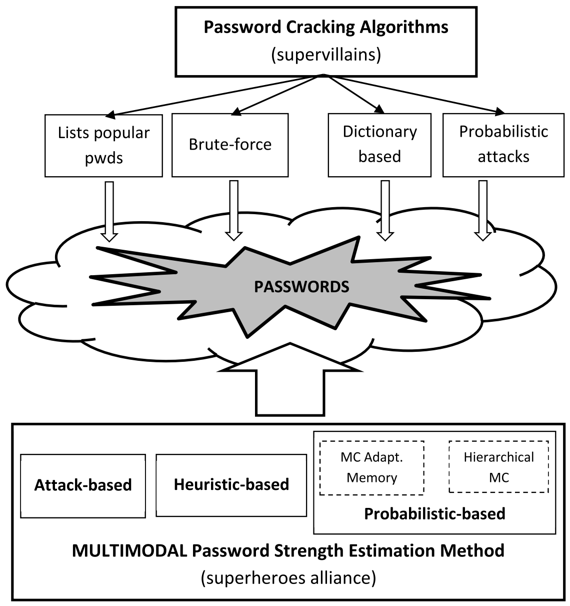

Fig. 3. General diagram showing the core idea behind the proposed multimodal password strength metric: The combination of several individual methods, each of them specialized at detecting a particular group of weak passwords, will lead to a more general and reliable overall approach.

strategy described in Sect. II, two types of passwords are differentiated: 1) trivial passwords and 2) non-trivial passwords. For each of the two types of passwords, a specific set of individual metrics has been developed to accurately estimate the strength of that particular group. The individual scores S_{AM} and S_N given by the strength modules designed to detect non-trivial passwords are normalized in an intermediate step, $\bar{S}_{AM}$ and $\bar{S}_N$, to the range of values [0,10]. This normalization step transforms the scores into one common domain and gives them a more easily interpretable meaning prior to their fusion (further details on the normalization and fusion are given in Sect. VII). Scores S_D and S_H generated by the modules focused on trivial passwords are directly produced in the range [0,10] and therefore do not need to be normalized. A final fusion module combines the normalized scores of each individual block to generate the definitive multimodal password strength value S_{MULTI}.

The four individual techniques that are fused into the final unique multimodal measure (see Fig. 4), have been developed to be highly flexible depending on a number of parameters that should be fixed on a case by case basis during an initial training phase. This way, the method meets the second of the objectives set out for the work: *adaptability*. This feature allows it to give more accurate strength estimations depending on each application-specific scenario. Essentially this means that the proposed metric can give different strength values to the same password depending on the context (i.e., training data). As pointed out in the introduction, this adaptability is a very desirable characteristic of strength metrics, since the guessability of one same password can vary significantly depending on the context where it is used.

The different modules shown in Fig. 4 are described in the following sections. For clarity, we introduce here some basic mathematical notation that will be used in the following sections: a generic password pwd will be noted as a chain of L characters c^l, that is, $pwd = (c^1, c^2, \dots c^l, c^{l+1}, \dots c^L)$, where the superindex l denotes the position of the character in the password; the characters are selected from an alphabet of size N.

V. Strength Module 1: Trivial Passwords

Any efficient strength metric should be able to detect trivial passwords that do not withstand the two basic attacks that will be performed almost with all certainty at the beginning of any password guessing session: attacks based on a list of the most used passwords and brute-force attacks (see Sect. II). Such passwords should be assigned a very low strength score as essentially there is the guarantee that they will be broken. The present strength module, composed of two submodules, is designed to detect these very weak trivial passwords whose final strength should be equal or close to 0.

As already mentioned in Sect. II, the classification between trivial and non-trivial passwords is not deterministic, it will depend on the application being considered. The two strength modules described next have been conceived from a general perspective so that, properly selecting their input parameters, they can adapt to different frameworks and applications.

A. Strength Module 1A: Attack-Based

In order to detect extremely common passwords, the present strength module 1A is introduced in the overall system. The module is based on blacklisting, which has now been shown in different works to be a highly valuable method against password cracking attacks [31], [54]. The objective of this metric is to detect the very simple, but very effective, attack used by an adversary that seeks to gain access by just trying a list of common passwords [34]–[36]. Such attack is specially relevant in a scenario where intruders try to get illegal access to an on-line application for which a *very limited* number of attempts (i.e., guesses) per time unit is allowed. In this very constrained situation in terms of access attempts, adversaries will try to break the system by simply trying the known “worst of the worst passwords” (i.e., most common passwords).

To increase the generality of the module, that is, to avoid making it fully dependent on the selected list of passwords, any test password with a Levenshtein distance [66] equal or lower than 1, i.e., $LD \leq 1$, to any password in the list, is considered to be of very low strength. In essence, this means that any test password with a difference of just one single-character edit (i.e., insertion, deletion or substitution) with respect to a password in the list is assumed to be a trivial password.

Accordingly, based on a password list $List_{pwd}$ (input parameter), the module takes a test password pwd and returns a strength value S_D depending on whether the password complies with $LD(List_{pwd}, pwd) \leq 1$. As such, only two strength values are possible: $S_D = 0$ for any of the most popular passwords or 1 character variations of them; and $S_D = 10$ for those that do not fall in the previous category. No distinction is made among passwords that are robust to the attack as they all receive the same high score (that may be later lowered by the rest of the strength modules implemented in the overall method).

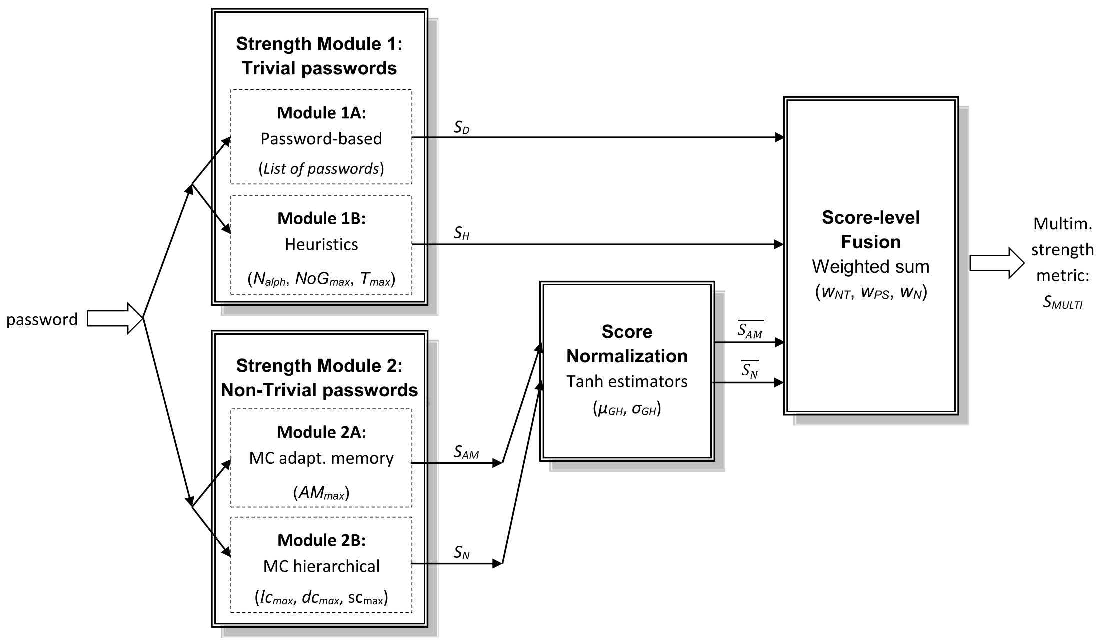
Strength Module 1: Trivial passwords
Module 1A: Password-based (List of passwords)
Module 1B: Heuristics (N_{alph}, NoG_{max}, T_{max})
password
Strength Module 2: Non-Trivial passwords
Module 2A: MC adapt. memory (AM_{max})
Module 2B: MC hierarchical (lc_{max}, dc_{max}, sc_{max})
S_D
S_H
S_{AM}
S_N
Score Normalization Tanh estimators (μ_{GH}, σ_{GH})
$\overline{S_{AM}}$
$\overline{S_N}$
Score-level Fusion Weighted sum (w_{NT}, w_{PS}, w_N)
Multim. strength metric: S_{MULTI}

Fig. 4. General diagram of the proposed multimodal strength metric described in Sect. IV. For each module, the main parameters are specified in brackets.

The metric is highly dependent on the size of the list of passwords considered. Such list is a design parameter that can be changed or updated by the service provider in order to reach a compromise between: security *vs* convenience. If a very large list of banned passwords is included (e.g., in the range of millions) the system will certainly be more secure but the frustration of the users at the moment of selecting the password will most likely also increase.

Different works have studied the impact of blacklists size on the security of the system [8], [31], showing the advantages of using significantly extensive lists. Against these recommendations, for the novel multimodal strength metric, it is advised the use of a quite restricted list since the present attack-based module does not work on its own, but it is complemented by other algorithms that can effectively detect weak word-based passwords, as is in particular the case of the Markov Chain with Adaptive Memory described in Sect. VI-A. This way, it is recommended to use this module for the detection of "the worst of the worst" passwords, while leaving other "not-so-critic" cases to be ranked by the rest of the modules.

Module 1A parameters: Following the description given above, the parameters of the password-based module that need to be specified at the training phase of the overall method, are: list of banned passwords.

B. Strength Module 1B: Heuristic-Based

This module takes as input a password and returns a strength value S_H based on its complexity. It is specifically designed to detect those passwords that are brute-forceable (i.e., that may be broken by means of a brute-force attack) in a given scenario. As such, only two strength values are possible. All passwords that are not complex enough and that, therefore, would not resist a brute-force attack, are given the same very low score $S_H = 0$. On the other hand, all passwords that are robust to brute-force are assigned the same very high score $S_H = 10$ (that may be later lowered by the rest of the strength modules implemented in the overall method).

As already mentioned in Sect. III, a standard brute-force attack consists of trying all possible character combinations (i.e., passwords) within a given search space until the targeted password is retrieved.

Nowadays it is accepted that, given the current state of development of computing technology, passwords below a certain complexity (i.e., in terms of length L and number of symbols N in the alphabet used) can be broken by means of the brute-force strategy described above (i.e., exhaustive search in the password space). The size of such password space is defined by the password complexity, and is equal to N^L.

The challenge in this case is to define what a brute-forceable password is for each different application/scenario. That is, what is the minimum complexity in terms of N and L that a password must present not to be found by means of a brute-force attack given some application-specific implementation constraints.

It should be noticed that the number of alphabet symbols N defining the search space is defined by the password being evaluated itself. For example, in the case of a password with only lower-case letters $N = 26$, while in the case of a password with lower-case and upper-case letters $N = 52$. This way, the complexity parameter to be determined on a case by case basis is the minimum length L_{min} that passwords should have in order to withstand an exhaustive search given that specific N. This minimum length depends on two factors:

- Maximum number of guesses per second NoG_{max}. This value may be computed as $NoG_{max} = \min[NoG^{app}_{max}, NoG^{tech}_{max}]$, that is, the minimum between: (i) NoG^{app}_{max}, the maximum number of guesses per second allowed by the application being attacked, which depends, among others, on the type of encryption strategy and oracle being used. Therefore, this is a

design parameter which can be defined by the service provider. (ii) NoG^{tech}_{max}, maximum number of guesses allowed by the technology used for the attack, which depends for instance on the computation capacity of the hardware platform at the disposal of the attacker (e.g., CPU, FPGA, GPU).
In the case of remote authentication services, as it is not possible to know what technology will be used to carry out a brute-force attack, a valid assumption is that it will be powerful and fast enough to perform all the guesses per second allowed by the service, that is, $NoG^{app}_{max} < NoG^{tech}_{max}$. In this common scenario, the equality $NoG_{max} = NoG^{app}_{max}$ will hold.

- Maximum time T_{max} allowed to run a brute-force attack. This is a design value to be determined by the service provider depending on the security level that he wants to offer. A higher T_{max} value will imply a higher security for the selected passwords.
As a concrete example, in the case of systems that include a policy defining a password renewal period, a sensible strategy to select T_{max} would be to set it to the password expiration time. That is, if users of a company's intranet must change their password every three months, T_{max} would be set to this same value. This way, the system designer ensures that the average time required by an eventual brute-force attack to break a given password is longer than the validity period of that password, and therefore the application is robust to that threat.

Given the three parameters defined above, N, NoG_{max} and T_{max}, selected by the system designer, it holds that: $NoG_{max} \times T_{max} = N^{L_{min}}$. Therefore, the minimum length L_{min} of non brute-forceable passwords can be estimated as:

$$L_{min} = \frac{\log\left(NoG_{max} \times T_{max}\right)}{\log\left(N\right)}. \quad (1)$$

This is the minimum length required so that passwords generated from an alphabet of N symbols are not systematically broken by a brute-force attack running for T_{max} time with a NoG_{max} computational power.

It can be assumed that, under these parameters, the probability that passwords shorter than L_{min} are guessed is equal to 1, whereas the probability that longer passwords are broken by means of a brute-force attack is very close to 0 (there is always a non-zero probability that a password is guessed by pure chance). As a consequence, the proposed strength metric will assign the minimum strength value $S_H = 0$ to all passwords shorter than L_{min}, and the highest strength value $S_H = 10$ to all longer passwords, independently of the password selected.

Module 1B parameters: Following the description given above, the parameter of the heuristics-based module that need to be specified at the training stage of the overall method, is: L_{min}.

VI. Strength Module 2: Non-Trivial Passwords

This strength module is designed to cope with passwords that have been assigned the highest score value by the previous modules 1A and 1B, i.e., non-trivial passwords that are robust to attacks based on lists of common passwords and to brute-force attacks.

As introduced in Sect. II, it is not possible to foresee all the specific attacks that will be performed to break these non-trivial passwords, therefore, as argued in Sect. III, the most sensible approach to assign them a strength value is to estimate the likelihood that a person would choose them, since there are already attacks that model this human behaviour [19], [48]. For this purpose, two novel probabilistic methods inspired in human password-selecting behaviour have been developed: 1) the Markov Chain with adaptive memory AM (described in Sect. VI-A); and 2) the Markov Chain based on hierarchical chains (described in Sect. VI-B).

Both methods are modified versions of the original Markov model concept [49]. Apart from many other problems related to machine learning and pattern recognition such as speech recognition [16], Markov models have been extensively used for language modeling [61], [67], [68], which is a field of research tightly related to password representation.

If the reader is not fully familiar with Markov Chains and other related concepts such as *smoothing*, it is highly recommended to go over some general work in the topic like [16], [49], [69] before moving to the description of the two novel methods. As an introduction to how Markov models can be applied to the representation of passwords, it is also important to get acquainted with the two simple models that were introduced in [19] and further analyzed in [62]: the discrete time Simple Markov Chain and the discrete time Layered Markov Chain. We believe that this initial introductory readings can help to better understand the more complex models developed in the work and presented in Sects. VI-A and VI-B.

Essentially, the objective of all the statistical models presented hereinafter is to assign a probability of occurrence po_{c^l} to each character c^l that composes a given password, so that, in the end, the probability of occurrence po_{pwd} of the whole password $pwd = \left(c^1, c^2, c^3, \ldots c^L\right)$ may be computed as the product of the probability of occurrence of all characters, i.e., $po_{pwd} = \prod_{l=1}^{L} po_{c^l}$.

A. Module 2A: Adaptive Memory Markov Chain

This module takes as input a password and returns a strength value S_{AM} based on a novel statistical model named *Markov Chain with adaptive memory*. It is specifically designed to detect local patterns in passwords derived from the use of known words.

As passwords are heavily influenced by language [44], we know that the transition probability to move from current state 'c' to next state 'a' is much higher if the sequence of precedent states was (R, e, b, e, c, c), than if the sequence of precedent states was (A, d, v, a, n, c), where the highest transition probability would be from the state 'c' to the state 'e'. This argumentation shows the need to consider statistical models with memory for the representation of human-selected passwords, such as AM-th order Markov Chain, a.k.a. AM-gram model [68], [70], that takes into account the AM previous characters. Such approach is extensively used in the field of computational linguistics, and already considered with some success for password strength estimation in [17].

Although this basic approach addresses many of the shortcomings of the Simple and Layered Markov Chains (see [62]), it introduces a new key issue in the representation of passwords: what would be the optimal memory size? The biggest memory possible could be a good choice, however, this would entail a huge model impossible to be properly trained with a finite (but very large) amount of data. Accordingly, the transition matrix would be highly sparse and its statistical significance very low. In addition, if the memory size is too big, the model would become equivalent to simply checking if the test password is in the training set of passwords. Therefore, it would cause the almost total loss of generality of the model, which would be unable to correctly evaluate passwords not present in the training set.

To address such lack of generality, we define a new approach using an adaptive memory size. In essence, the model considers as the current state of the Markov Chain the largest possible sub-sequence of AM characters (AM-gram) for which a transition to the following character exists in the transition matrix. If there is no such transition, the size of the memory is reduced by one character until a non-null transition is found. In those cases, a "bonus" is given to the evaluated password as we assume that a password that contains a sequence of AM characters previously observed in the training set is probably weaker than a password for which only a sequence of $AM - 1$ characters has been observed. Accordingly, a reduction in the memory should entail a decrease in the probability of occurrence of the password which is achieved through this bonus.

The model is formally defined by S states, where $S = (N+3)^{AM_{max}}$ and by the conditional transition probabilities $pt_{\alpha_i\alpha_j}$, where $(i, j) \in \{0, \dots, S-1\}^2$. Each state is a tuple of AM_{max} characters c taken from the set of $N + 3$ characters, that are the N characters from the alphabet plus three artificial characters: an *initialization character* c_0, an *undefined character* c_u, and an *end of password character* c_e. A state is denoted $\alpha_i = (c_i^1, c_i^2, \dots, c_i^{AM_{max}})$, for $i \in \{0, \dots, S-1\}$, and by convention $\alpha_0 = (c_0, \dots, c_0)$. The transition probabilities from one state to another have the two following properties:

1) The sum of the transition probabilities from a state α_i, for any $i \in \{0, \dots, S-1\}$ to any other state in the chain (including himself) must be equal to 1, that is $\sum_{j=0}^{S-1} pt_{\alpha_i\alpha_j} = 1$;
2) The transition probability from one state $\alpha_i = (c_i^1, c_i^2, \dots, c_i^{AM})$ to another state $\alpha_j = (c_j^1, c_j^2, \dots, c_j^{AM})$ can be greater than 0 only if $(c_i^2, \dots, c_i^{AM}) = (c_j^1, \dots, c_j^{AM-1})$. This represents the transition probability from a sequence of AM characters to a single character.

To evaluate the score S_{AM} of a password in the adaptive memory Markov chain model, all the transition probabilities of the sequence of states composing the password are evaluated, starting from the initial state α_0 to the last one containing the end of password character c_e. When a transition from one state to another is equal to zero, the first character of the considered state is replaced by the undefined character c_u, virtually reducing the size of the memory. The score of the password is multiplied by a correction coefficient *CrCoef*, with *CrCoef* < 1, corresponding to the bonus attributed to the password strength value previously mentioned. *CrCoef* is defined as the ratio between: the minimum non-zero probability of moving from the given sequence of characters of size AM to any individual character; and the maximum non-zero probability of moving from the sequence of characters of size $AM - 1$ (after the memory reduction) to any individual character. This process is repeated until a transition is found or eventually the length of the sequence is reduced to a single character, which is equivalent to the Simple Markov Chain.

Algorithm 1 Scoring Algorithm of the Adaptive Markov Chain Model

```
Input: The password to evaluate $pwd = (c^1, c^2, \dots, c^L)$
Result: The score $S_{AM}(pwd)$ of $pwd$
/* $(c_*)^{(k)}$ is a sequence of $k$ times $c_*$ */
$po_{pwd} = 1$; $\alpha^{(0)} = (c_0)^{(AM_{max})}$;
for $l = 0$ to $L$ do
    if $l = L$ then $\alpha^{(l+1)} = \alpha^{(l)}[1:]||c_e$ else
    $\alpha^{(l+1)} = \alpha^{(l)}[1:]||c^{l+1}$ no_transition = True; i=0;
    while no_transition do
        if $pt_{\alpha^{(l)}\alpha^{(l+1)}} != 0$ then
            no_transition = False;
            $po_{pwd} = po_{pwd} \cdot pt_{\alpha^{(l)}\alpha^{(l+1)}}$;
        else
            i=i+1;
            $po_{pwd} = CrCoef \cdot po_{pwd}$;
            $\alpha^{(l)} = (c_u)^{(i)}||\alpha^{(l)}[i:]$;
        end
    end
$S_{AM}(pwd) = -10\log_{10}(po_{pwd})$;
```

The final score S_{AM} attributed to a password is equal to $S_{AM} = -10\log_{10}(po_{pwd})$ and is computed as described in Algorithm 1.

Although the Adaptive Memory Model addresses the main limitation of the model with fixed memory, the issue of determining the maximum memory size AM_{max} still remains. There is not a unique optimal value that may be computed in a deterministic way. Rather, its estimation should be done heuristically on a case by case basis. To do so it should be noticed that, essentially, the transition matrix T is equivalent to an exhaustive combination table in a search space defined by AM_{max} and N (i.e., all possible AM_{max} character combinations taken from a pool of N characters are reflected in the matrix). As the training data is limited, the larger AM_{max}: 1) the more zeros will populate the table; and 2) the fewer number of observations that will be used to compute the probability of non-zero occurrence sequences. In summary, for a finite and limited set of training data, the larger AM_{max}, the lower the statistical reliability of the very sparse matrix T.

Therefore, two linked factors should be taken into account in order to select AM_{max}:

- Size of the training set: as a general rule, larger training sets will allow reliably training models defined by larger values of AM_{max}.
- Size of T: the larger AM_{max}, the bigger the transition matrix T, eventually requiring a very large storing capacity and also slowing down the strength estimation process. This issue can be partially mitigated in practice using a sparse matrix as, for long AM sequences, most transition probabilities are equal to zero.

Smoothing: Smoothing is only performed on the elements of the transition matrix corresponding to the Simple Markov Chain, that is, from a state containing one character of the alphabet to another single character. The additive smoothing algorithm first introduced in [71] is applied.

Transitions from states containing two or more characters of the alphabet are not smoothed as the memory reduction strategy followed by the model can be considered, in itself, a smoothing technique: transitions from shorter sequences are represented by smaller transition matrices that, as such, can be more accurately estimated on the same amount of training data (which is the ultimate goal of smoothing).

The discussion above is inspired by the Jelinek-Mercer smoothing that exploits, in the field of language modeling, a very similar principle to the one used here [72]: in general, it is useful to combine higher-order AM-gram models (i.e., Markov Models with fixed memory AM) with lower-order AM-gram models, because when there is insufficient data to estimate a probability in the higher-order model, the lower-order model can often provide useful information.

Advantages and Limitations of the Adaptive Memory Markov Chain: The Adaptive Memory Markov Chain presents two clear advantages: 1) on the one hand, being a memory model, it represents passwords in a more realistic way, where each character does not depend solely on the previous one. This way, it is and efficient model to detect dictionary-like passwords that, in general, are quite weak. 2) On the other hand, thanks to its adaptive memory, it does not lose generality with respect to the original Simple Markov Chain.

In spite of its strengths, the Adaptive Memory Markov Chain still has some limitations: 1) The model does not exploit sufficiently the overall structure of passwords. That is, characters in passwords are not placed randomly, rather, they present certain general patterns according to their class [11], [54] (i.e., letters, digits or special characters). 2) The size of the model can already become an issue for relatively small values of AM_{max}.

Module 2A parameters: Following the description given above, the parameters of the Adaptive Memory Markov Chain module that need to be specified during the training of the method are: AM_{max} and the transition matrix T.

B. Module 2B: Hierarchical Markov Chain

This module takes as input a password and returns a strength value S_N based on a novel statistical model named Hierarchical Markov Chain. It is specifically designed to detect the global structure of human-selected passwords.

The previous section has shown that, although the model with adaptive memory has clear advantages with respect to previous probabilistic-based methods such as the ones proposed in [17], [19], [62], it still presents some limitations.

As an attempt to address the previous issues, a new Markov Chain based on a hierarchical architecture is presented. Up to this point, all models have considered characters as the only constituent units of passwords. As such, their main focus has been the estimation of the individual characters probability of occurrence in order to, based on those individual probabilities, infer the likelihood of the complete password.

Only the Adaptive Memory model has considered forming elements in passwords larger than unique characters (i.e., sequences of characters). However, this constitutes a quite basic first step, as it takes into account specific character combinations and not a whole character class. That is, it computes the probability of occurrence of, for instance, the specific subsequence “vid”, but it does not consider the probability of occurrence of the general class “any combination of 3 lowercase letters”.

The Hierarchical Markov Chain is based on the hypothesis that: the characters of human-selected passwords are not grouped randomly. Instead, they join following certain rules to form larger structures that confer passwords recognizable patterns. The more a password is aligned with these overall rules, the more likely that it will be selected by a human and, therefore, the weaker it is. This same hypothesis is the basis of the Probabilistic Context-Free Grammar (PCFG) approach initially used for password cracking described in [48].

The previous assumption implies that, although the fundamental password units are the characters, these group together in higher-order elements that, in turn, group together to form the password. In this frame, the Hierarchical Markov Chain is a tree-like model that first estimates the probability of occurrence of the higher-order structures within the password and then, in a subsequent step, estimates the probability of occurrence of characters within each structure. As such, the states of the higher order Markov Chain are defined by another (nested) Markov Chain.

In particular, the higher-order elements considered in the model are the next SS subsequences divided in three classes:

- Letter-class: formed by letter subsequences of sizes 1 to lc_{max}.
- Digit-class: formed by digit subsequences of sizes 1 to dc_{max}.
- Special characters-class: formed by special character subsequences of size 1 to sc_{max}.

In this model a generic password is formed by a sequence of P higher-order elements taken from the SS subsequences belonging to the three classes defined above, that is $pwd = \left(ss^1, ss^2, \dots ss^p, \dots ss^P\right)$. In turn, each of those higher-order elements is formed by R^p characters, that is $ss^p = \left(c^{p,1}, c^{p,2}, \dots c^{p,r}, \dots c^{p,R^p}\right)$, where $c^{p,r}$ represents the r-th character of the p-th subsequence.

The probability of occurrence of higher-order structures is defined as po_{ss^p}, while the probability of occurrence of characters within each higher-order structure is defined as $po_{c^{p,r}}$. This way, the probability of occurrence of a generic password $pwd = (ss^1, ss^2, \dots ss^p, \dots ss^P)$ can be computed as:

$$po_{pwd} = \prod_{p=1}^{P} \left[po_{ss^p} \left(\prod_{r=1}^{R} po_{c^{p,r}} \right) \right]. \tag{2}$$

As in the case of the Adaptive Memory model, the final strength metric S_N assigned to the password is finally computed as $S_N(pwd) = -10\log_{10}(po_{pwd})$, so that the less likely the password is according to the model, the stronger it is according to the metric.

The probability of occurrence of higher-order structures po_{s^p} is computed according to the Layered Markov Chain described in [62], where each state is assigned one of the SS higher-order subsequences. Any transition to/from a subsequence longer than lc_{max}, dc_{max} or sc_{max}, is assigned a probability equal to the same transition to/from the longest subsequence of its class divided by 10.

Then, each higher-order structure is modeled with a different Simple Markov Chain (see [62]) that outputs the probability of occurrence $po_{c^{p,r}}$ of the characters within each particular subsequence. Each state of the simple chain is assigned one of the N valid characters for the generation of subsequences of that specific class, that is: $N = 52$ ASCII capital and lower-case letters for the letter-class, $N = 10$ ASCII digits for the digit-class, $N = 32$ ASCII visible special characters for the special character-class. Subsequences longer than lc_{max}, dc_{max} or sc_{max}, are represented by the Simple Markov Chain corresponding to the longest subsequence of its class.

As a simple example, according to the Hierarchical Markov Chain, password "Pet52!" is formed by $SS = 3$ higher order elements: $ss^1 =$ Pet, $ss^2 = 52$ and $ss^3 =$!. The occurrence probability of this password would be:

- The probability that a password formed by 3 higher-order elements starts with a subsequence of 3 letters, multiplied by the probability that the first letter of a 3-letter subsequence is "P", multiplied by the probability that in a 3-letter subsequence "P" is followed by "e", multiplied by the probability that in a 3-letter subsequence "e" is followed by "t", multiplied by the probability that
- in a password with 3 higher-order elements a 3-letter subsequence is followed by a 2-digit subsequence in the second position, multiplied by the probability that the first digit of a 2-digit subsequence is "5", multiplied by the probability that in a 2-digit subsequence "5" is followed by "2", multiplied by the probability that
- in a password with 3 higher-order elements a 2-digit subsequence is followed by a 1-special character subsequence in the third position, multiplied by the probability that the first special character of a 1-special character subsequence is "!".

Smoothing: The smoothing technique used for all transition matrices involved in the model is the additive smoothing introduced in [71].

Advantages and Limitations of the Hierarchical Markov Chain: Given the definition above, the Hierarchical Markov Chain presents some noticeable advantages as it is able to model: 1) the general structure of passwords, like for instance, the order in which the considered character subsequences appear (e.g., digits are usually placed at the end); 2) the likelihood that characters of the same class (i.e., letters, digits, special characters) are grouped together; 3) what characters and transitions within each higher-order subsequence are more probable to happen.

In addition, the Hierarchical Markov Chain: 4) is a quite general model capable of representing a very large range of the password space (and not just the one containing the train set); 5) the model is still relatively compact and can be accurately trained with a reasonable amount of data.

In spite of the previous strengths, the Hierarchical Markov Chain is a model without memory, and therefore, it does not detect the occurrence of specific local character sequences (usually word related), that, as explained in the Adaptive Memory model, are common in weak passwords vulnerable to simple dictionary attacks.

Module 2B parameters: Following the description given above, the parameters of the Hierarchical Markov Chain module that need to be specified as part of the training phase of the overall method are: lc_{max}, dc_{max}, sc_{max}, parameter P_{max} (from the Layered Markov Chain), $P_{max}(1 + P_{max})/2$ transition matrices T^p corresponding to the Layered Markov Chain, SS transition matrices T corresponding to the Simple Markov Chains modeling each of the considered subsequences.

VII. Normalization and Fusion Modules

The final objective of the normalization and fusion modules is to combine the scores provided by the four individual strength estimation algorithms into the final multimodal score.

One of the lessons learned in the field of information fusion is the "complementarity principle", that is, the combination of complementary systems measuring different properties of the same problem tends to provide a large performance gain. In this regard, given the descriptions of the Adaptive Memory Markov Chain and the Hierarchical Markov Chain provided in Sect. VI-A and Sect. VI-B, it may be concluded that the two models are complementary, as they are designed to capture different aspects of human selected passwords: the Adaptive Memory Markov Chain can be considered as a *local model* that searchers for specific word-related patterns, while the Hierarchical Markov Chain may be understood as a *global model* that accurately represents the general structure of human passwords. The complementarity of the two proposed probabilistic models is experimentally analyzed in Part II of this series of two papers [18].

Therefore, following the information fusion "complementarity principle", it seems reasonable to assume that a multimodal strength metric could largely benefit from the output of the two models. To do so, the present fusion module uses a score-level combination strategy which is explained in the following.

A general theoretical framework for merging the scores obtained from multiple classifiers was presented in [73]. That work describes different combination algorithms that have been later evaluated in diverse information fusion experimental frameworks. One of the fusion approaches that has consistently showed high performance in very different studies is the *weighted sum* [74], which combines the scores s_k from K sources into one final score S_{MULTI} according to the linear equation:

$$S_{MULTI} = \sum_{k=1}^{K} w_k s_k, \tag{3}$$

where w_k are the weights given to each of the scores, generally complying with $\sum_{k=1}^{K} w_k = 1$.

One of the challenges of fusion systems is that the scores from the individual sources can be very heterogeneous. For instance, they may not necessarily be on the same numerical scale or may follow different statistical distributions. Therefore, prior to combining them into a single multimodal score, they need to be transformed into one common domain. This is accomplished through a process known as *score normalization*, which plays a critical part in the design of a combination scheme for score level fusion.

Although many techniques can be used for score normalization, the challenge lies in identifying a technique that is both robust and efficient. Robustness refers to insensitivity to the presence of outliers. Efficiency refers to the proximity of the obtained estimate distribution to the optimal estimate when the real distribution of the data is known. Several of the most common normalization schemes were analyzed in [75], where the *tanh estimators*, first introduced in [76], were identified as one of the best strategies in terms of robustness-efficiency tradeoff. According to this algorithm, the normalized score $\bar{s_k}$ in the range [0,10] is given by:

$$\bar{s_k} = 5[\tanh(0.01(s_k - \mu_{GH})/\sigma_{GH}) + 1], \tag{4}$$

where μ_{GH} and σ_{GH} are the mean and standard deviation estimates, respectively, of the score distribution. These two parameters, μ_{GH} and σ_{GH}, are computed on a training set of scores as given by the Hampel estimators in [76].

Even though potentially any arbitrary range of values could have been selected for the normalization, the choice of [0,10] is motivated by two main factors: 1) On one hand, the ultimate goal of a strength metric is to be used and interpreted by humans, and humans are very familiar with ranks that go from 0 to 10. From our subjective perspective, it may be easier for a regular user to understand that the strength of his password is 2/10 than 0.2/1. 2) On the other hand, selecting a range such as [0,1], could erroneously lead to think that the strength value given by the method is a probability, which is not correct. The metric is based on probabilities, but it does not comply with the mathematical properties required to be a probability.

Taking into consideration the discussion above, in the current work, the tanh estimators are used to compute the normalized strength scores of the Adaptive Memory model $\bar{S_{AM}}$ and the Hierarchical Markov Chain $\bar{S_N}$ prior to their combination using the weighted sum (please see Fig. 4). The strength scores of the trivial password modules S_H and S_D, as defined in Sects. V-A and V-B, can only take the values 0 or 10 and therefore do not need to be normalized. The final multimodal score is finally computed following equation 3 as:

$$S_{MULTI} = w_T \cdot (S_H \cdot S_D) + w_{AM} \cdot \bar{S_{AM}} + w_N \cdot \bar{S_N}. \tag{5}$$

The three weights $[w_T, w_{AM}, w_N]$ comply with $w_T + w_{AM} + w_N = 1$, and define the importance of each of the individual scores in the final multimodal strength indicator.

Parameters of the Normalization and Fusion modules: Following the description given above, the parameters of the normalization and fusion modules that need to be specified as part of the training phase of the overall method, are: μ_{GH} and σ_{GH} for the normalization, and the three weights $[w_T, w_{AM}, w_N]$ for the fusion.

VIII. Discussion: Contributions With Respect to Previous Approaches

Given the significant amount of password strength metrics proposed in the literature (see Sect. III for a non-comprehensive list), at least two different platforms have been recently announced that integrate some of the most relevant methods available: 1) "Password Analysis and Research System - PARS" which also includes several cracking algorithms [77]; 2) "Password multi-checker" which includes several password checkers from eleven famous web-services like Dropbox, Yahoo or Skype [78]. Undoubtedly these are very relevant initiatives for the field of password security, however, they do not propose new individual metrics, neither they consider the combination of different metrics into a single multimodal one. The contributions of those works are more related to the generation of a common framework for the performance evaluation of password strength algorithms. To the best of our knowledge, the multimodal strength metric proposed in the present article, based on the principle of complementarity of single algorithms, is the first of its kind.

Nevertheless, the four individual modules that conform the overall method do present similarities with previously proposed techniques, in particular: 1) Modules focused on trivial passwords: The attack-based and heuristic-based modules (described in Sects. III-A and III-B) present very limited novelty with respect to many of the works presented in the review of the state of the art, in Sect. III-A and Sect. III-B, respectively. 2) Modules focused on non-trivial passwords: The two probabilistic-based methods do present notable novelty with respect to previous related approaches mentioned in the review of the state of the art in Sect. III-C. The contributions of each of these particular algorithms are highlighted in the following subsections.

A. Contributions: Markov Chain With Adaptive Memory

The most related work from the state of the art to the Adaptive Memory Markov Chain is the one by Ma *et al.* [15]. Inspired by Katz's Language model [79], they have also used Markov chains of variable orders in their comparative study of probabilistic password models. In their approach, both the previous sequence of characters and the ending character are taken into consideration. The memory is not bound by an upper limit but by the frequency of appearance of the considered subsequence (i.e., AM-gram in our model). If it is lower than a predefined threshold then the memory is reduced and the updated transition is considered. A smoothing function is applied on the probability obtained so that the sum of the probabilities for all subsequences of a given length remains equal to one.

As detailed in their paper, the probability estimation is significantly slower than all the other methods they present. Although it is not specified in their article, this is probably due to the impossibility to fully train a Markov model considering any memory size, thus forcing to estimate the transition probabilities while computing the strength metric. This makes the model virtually unusable for real-time strength estimation.

In the case of the adaptive memory method described in the present section, the maximum memory size AM_{max} is fixed in order to define a transition matrix that can be accurately estimated on the available training data. This way, there is no need to compute transition probabilities at the time of password evaluation and therefore can be used in real time. The memory reduction strategy followed in our model also decreases the sparseness of the data, allowing to train the model on a dataset more than 100 times bigger than the one considered in [15]. In summary, it may be stated that the current approach is a feasible, practical implementation.[2] of the more theoretical model presented in [15]

B. Contributions: Hierarchical Markov Chain

The Hierarchical Markov Chain was inspired by the first work in the field of probabilistic-based models that contemplated the existence of higher-order structures presented in [48] and further developed in [63], always in the field os password cracking. That pioneering study considered that passwords are formed by character subsequences combined following certain rules, as defined by Probabilistic Context-Free Grammars (PCFG). In a way, the author defines a password-specific grammar for human-selected passwords. Although certain similarities exist between the two models, the hierarchical MC goes beyond its predecessor trying to give an answer to some of its limitations, by combining the high potential of PCFG for modeling structured languages, with the power of Markov Chains for building symbol strings following a certain observation probability. The main differences between both models are summarized in the following.

Regarding higher order structures, the model proposed in [48] considers only the probability of appearance of fixed subsequence-combinations. That is, if a password formed by the *exact* subsequence combination "L7D3S2" was never observed in the train set (where L7 is a subsequence of 7 letters, D3 a subsequence of 3 digits and S2 a subsequence of 2 special characters), it is impossible that it is produced by the model.

The Hierarchical Markov Chain is more flexible and not so training-data driven, as it does not represent specific subsequence-combinations, but transition probabilities between subsequences. As such, it can estimate the probability of occurrence of the sequence "L7D3S2" even if it never appeared in the train set. Let's assume, for instance, that the training set contained the sequences "L7D1S1", "L7D3L2" and "S1D3S2". Then, the probability assigned to sequence "L7D3S2" will be: the probability that 'L7' is the first subsequence of a password formed by 3 subsequences, multiplied by the probability that 'L7' is followed by 'D3' in a 3-subsequence password, multiplied by the probability that 'D3' is followed by 'S2' in a 3-subsequence password. Since none of those probabilities is zero, the sequence "L7D3S2" is a possible output of the Hierarchical Markov Chain even though not previously seen.

Regarding lower-order structures, the model proposed in [48] represents only those specific character combinations that appeared in the training set. That is, if the 7-letter subsequence L7='mYpEtyy' was never observed in the training word dictionary, it will not be produced by the model (i.e., it will be assigned a zero probability of occurrence).

Since the Hierarchical Markov Chain represents each subsequence with a Simple Markov Chain, it is totally general and is capable of modeling subsequences that never appeared in the training set. Continuing with the previous example, it is able to assign a probability of occurrence to L7='mYpEtyy' as long as the training set contains different 7-letter subsequences where: 'm' is the first letter, 'm' is followed by 'Y' (at any position), 'Y' is followed by 'p' (at any position), etc.

In summary, although both models are based on the existence of higher-order patterns, the proposed Hierarchical Markov Chain is an evolution of the probabilistic PCFG model as it presents a significantly higher generality, being able to model password structures and character combinations not observed in the training set.

IX. Conclusions

Traditional password strength metrics are becoming inefficient against the new generation of most advanced password guessing attacks that are being used against real applications. In this context, new and more reliable approaches for the estimation of password robustness are required to protect users against potential external threats as shown by the drastic change of direction with respect to previous versions of NIST's latest draft of recommendation SP 800-63-3 [59].

Different works have shown the inefficiency of current methodologies used by service providers to persuade users to select stronger passwords. This includes traditional password composition policies [54], [80], oriented only to avoid the selection of trivial passwords vulnerable to brute-force attacks, but that do not consider other type of common attacks such as the ones using wordlists of common passwords. Furthermore, some passwords that do not satisfy those imposed requirements can be highly secure, such as for instance a 20-character password with just lower-case letters.

In this context, new and more effective password strength metrics are required to support reasonable password policies and to estimate password security more accurately. Following this objective, the present work has described a novel multimodal and adaptable method for the estimation of password strength. The main rationale behind the proposed approach is to exploit the capabilities of different individual techniques

[2]Part II of this series of papers is accompanied by the executable graphical application JRC-PaStMe that integrates an implementation of this method.

and, through their fusion, achieve one unique multimodal measure which overcomes many of the weaknesses of the unimodal methods. We believe that such approach is, for instance, largely aligned with the new philosophy in password security drafted by NIST [59].

To achieve the goal of multimodality and adaptability, two new probabilistic methods based on the popular Markov Chains have been proposed to accurately estimate the strength of non-trivial passwords: 1) the Markov Chain with adaptive memory and 2) the Hierarchical Markov Chain. The two models are complementary as they focus on capturing different aspects of human selected passwords: the Adaptive Memory Markov Chain can be considered as a *local model* that searchers for specific word-related patterns, while the Hierarchical Markov Chain may be understood as a *global model* that accurately represents the general structure of passwords.

Taking advantage of the knowledge gained in the field of machine-learning multimodal systems, the two previous statistical methods are fused with an attack based module (using blacklisting) and an heuristic-based module, both specifically designed to detect trivial passwords. The result is a single multimodal strength score which condenses the information extracted from each of the four individual models.

The new multimodal-adaptable method may be a valuable tool both for service providers and end-users. On the one hand, service providers can use it as a security control to enforce the adoption of strong passwords, by integrating it in the password selection procedure in such a way that users are not allowed to choose weak passwords, or at least warned when they do so. On the other hand, end-users will receive a real-time feedback concerning the likelihood that their password would be broken should there be an attack or a leakage.

In Part II of the present work [18], we describe the experimental framework followed to validate the proposed approach, where it is shown that the multimodal metric produces sensible strength values fully correlated with the resistance of passwords to attacks of increasing complexity.

REFERENCES

[1] B. News. (May 2016). *Millions of Hacked LinkedIn IDs Advertised for Sale*. [Online]. Available: http://www.bbc.com/news/technology-36320322

[2] KoreLogic. (May 2016). *LinkedIn Revisited—Full 2012 Hash Dump Analysis*. [Online]. Available: https://blog.korelogic.com/blog/2016/05/19/linkedin-passwords-2016

[3] The Telegraph. (Jun. 2016). *Mark Zuckerberg's Password Was 'Dadada What Hope do The Rest of us Have?*. http://www.telegraph.co.uk/technology/2016/06/06/mark-zuckerbergs-passw%ord-was-dadada-what-hope-do-the-rest-of-us/

[4] C. Herley and P. C. van Oorschot, "A research agenda acknowledging the persistence of passwords," *IEEE Security Privacy*, vol. 10, no. 1, pp. 28–36, Jan. 2012.

[5] M. Bishop and D. V. Klein, "Improving system security via proactive password checking," *J. Comput. Secur.*, vol. 14, no. 3, pp. 233–249, 1995.

[6] S. Furnell, "An assessment of website password practices," *Comput. Secur.*, vol. 26, nos. 7–8, pp. 445–451, 2007.

[7] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang, "The tangled Web of password reuse," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2014, pp. 23–26.

[8] P. G. Kelley, S. Komanduri, M. L. Mazurek, R. Shay, and T. Vidas, "Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms," in *Proc. IEEE Symp. Secur. Privacy (SSS)*, May 2012, pp. 523–537.

[9] D. Wheeler. (2012). *Zxcvbn: Realistic Password Strength Estimation*. [Online]. Available: https://blogs.dropbox.com/tech/2012/04/zxcvbn-realistic-password-streng%th-estimation/

[10] J. Bonneau, "The science of guessing: Analyzing an anonymized corpus of 70 million passwords," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2012, pp. 538–552.

[11] R. Shay *et al.*, "Encountering stronger password requirements: User attitudes and behaviors," in *Proc. USENIX Symp. Usable Privacy Secur. (SOUPS)*, 2010, p. 2.

[12] W. E. Burr *et al.*, "NIST special publication 800-63-2: Electronic authentication guideline," NIST, Gaithersburg, MD, USA, Tech. Rep. 800-63-2, 2012.

[13] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul./Oct. 1948.

[14] J. Bonneau, "Statistical metrics for individual password strength," in *Proc. ACM Int. Conf. Secur. Protocols (ICSP)*, 2012, pp. 76–86.

[15] J. Ma, W. Yang, M. Luo, and N. Li, "A study of probabilistic password models," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2014, pp. 689–704.

[16] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[17] C. Castelluccia, M. Durmuth, and D. Perito, "Adaptive password-strength meters from Markov models," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2012, pp. 1–14.

[18] J. Galbally, I. Coisel, and I. Sanchez, "A new multimodal approach for password strength estimation. Part II: Experimental validation," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 12, pp. 2845–2860, Dec. 2017.

[19] W. Tansey, "Improved models for password guessing," Univ. Texas, Austin, TX, USA, Tech. Rep., 2011. [Online]. Available: https://www.cs.utexas.edu/ tansey/passwords.pdf

[20] D. Seeley, "Password cracking: A game of wits," *Commun. ACM*, vol. 32, no. 6, pp. 700–703, 1989.

[21] B. Ur *et al.*, "I added ! at the end to make it secure: Observing password creation in the lab," in *Proc. USENIX Symp. Usable Privacy Secur. (SOUPS)*, 2015, pp. 123–140.

[22] D. Florencio and C. Herley, "A large-scale study of Web password habits," in *Proc. ACM Int. Conf. World Wide Web (WWW)*, 2007, pp. 657–666.

[23] B. Schneier. (2005). *Write Down Your Passwords*. [Online]. Available: https://www.schneier.com/blog/archives/2005/06/write-down-your.html

[24] J. Yan, A. Blackwell, R. Anderson, and A. Grant, "Password memorability and security: Empirical results," *IEEE Security Privacy*, vol. 2, no. 5, pp. 25–31, Sep./Oct. 2004.

[25] A. Adams and A. M. Sasse, "Users are not the enemy," *Commun. ACM*, vol. 42, pp. 40–46, Apr. 1999.

[26] J. Steube, "Introducing the PRINCE attack-mode," in *Proc. Int. Conf. Passwords*, 2014, pp. 1–42.

[27] OpenWall. *John-the-Ripper Open Source Password Cracker*, accessed on Jan. 9, 2017. [Online]. Available: http://www.openwall.com/john/

[28] M. Weir and S. Aggarwal, "Cracking 400,000 passwords or how to explain to your roommate why the power-bill is a little high," in *Proc. DefCon Conf.*, 2009.

[29] Y. Chrysanthou, "Modern password cracking: A hands-on approach to creating an optimised and versatile attack," M.S. thesis, Dept. Inf. Secur., Royal Holloway Univ. London, Egham, U.K., 2013.

[30] B. Ur *et al.*, "Measuring real-world accuracies and biases in modeling password guessability," in *Proc. USENIX Secur. Symp.*, 2015, pp. 463–481.

[31] M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, 2010, pp. 162–175.

[32] Wired. (2009). *Weak Password Brings 'Happiness' to Twitter Hacker*. [Online]. Available: http://www.wired.com/2009/01/professed-twitt/

[33] New York Times. (2010). *If Your Password is '123456', Just Make it 'HackMe'*. [Online]. Available: http://www.nytimes.com/2010/01/21/technology/21password.html?-r=0

[34] WhatsMyPass. (2008). *The Top 500 Worst Passwords of All Time*. [Online]. Available: http://www.whatsmypass.com/the-top-500-worst-passwords-of-all-time

[35] TechCrunch. (2009). *370 Passwords You Shouldn't (and Can't) Use on Twitter*. [Online]. Available: http://techcrunch.com/2009/12/27/twitter-banned-passwords/

[36] ReusableSecurity. (2009). *The RockYou 32 Million Password List Top 100*. [Online]. Available: http://reusablesec.blogspot.it/2009/12/rockyou-32-million-password-list%-top.html

[37] SplashData. (2016). *Announcing Our Worst Passwords of 2015*. [Online]. Available: https://www.teamsid.com/worst-passwords-2015/

[38] SkullSecurity. (2009). *Passwords*. [Online]. Available: https://wiki.skullsecurity.org/Passwords

[39] R. Morris and K. Thompson, "Password security: A case history," *Commun. ACM*, vol. 22, no. 11, pp. 594–597, 1979.

[40] P. Oechslin, "Making a faster cryptanalytic time-memory trade-off," in *Proc. Int. Conf. Adv. Cryptol. (CRYPTO)*, 2003, pp. 617–630.

[41] R. Project. (2015). *RainbowCrack*. [Online]. Available: http://project-rainbowcrack.com/

[42] M. E. Hellman, "A cryptanalytic time-memory trade-off," *IEEE Trans. Inf. Theory*, vol. 26, no. 4, pp. 401–406, Jul. 1980.

[43] OpenWall. (2015). *Openwall Wordlists Collection*. [Online]. Available: http://www.openwall.com/wordlists/

[44] R. Veras, C. Collins, and J. Thorpe, "On the semantic patterns of passwords and their security impact," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2014, doi: http://dx.doi.org/ndss.2014.23103

[45] CrackStation. (2016). *CrackStation: Free Password Hash Cracker*. [Online]. Available: https://crackstation.net/

[46] Hashcat. (2016). *Hashcat: Advanced Password Recovery*. [Online]. Available: https://hashcat.net/

[47] M. Dürmuth, F. Angelstorf, C. Castelluccia, D. Perito, and A. Chaabane, "OMEN: Faster password guessing using an ordered Markov enumerator," in *Proc. Int. Symp. Eng. Secure Softw. Syst. (ESSoS)*, 2015, pp. 119–132.

[48] M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek, "Password cracking using probabilistic context-free grammars," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2009, pp. 391–405.

[49] J. R. Norris, *Markov Chains*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[50] J. M. Gosney. (Jun. 2016). *8x Nvidia GTX 1080 Hashcat Benchmarks*. [Online]. Available: https://gist.github.com/epixoip/a83d38f412b4737e99bbef804a270c40

[51] S. Brostoff and A. Sasse, "'Ten strikes and you're out': Increasing the number of login attempts can improve password usability," in *Proc. CHI Workshop Human Comput. Interact. Secur. Syst.*, 2003. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.8.2500&rep=rep1&type=pdf

[52] M. L. Mazurek *et al.*, "Measuring password guessability for an entire University," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, 2013, pp. 173–186.

[53] R. Shay *et al.*, "Can long passwords be secure and usable?" in *Proc. SIGCHI Conf. Human Factors Comput. Syst. (CHI)*, 2014, pp. 2927–2936.

[54] R. Shay *et al.*, "Designing password policies for strength and usability," *ACM Trans. Inf. Syst. Secur.*, vol. 18, no. 4, 2016, Art. no. 13.

[55] D. L. Wheeler, "Zxcvbn: Low-budget password strength estimation," in *Proc. USENIX Secur. Symp.*, 2016, pp. 157–173.

[56] PasswordMeter. (2015). *The Password Meter*. [Online]. Available: http://www.passwordmeter.com/

[57] Microsoft. (2015). *Safety and Security Center: Password Checker*. [Online]. Available: http://www.microsoft.com/it-it/security/pc-security/password-checker.as%px

[58] XKCD. (Aug. 2011). *Password Strength*. [Online]. Available: https://xkcd.com/936/

[59] NIST. (Aug. 2016). *NIST SP 800-63-3: Public Preview*. [Online]. Available: https://pages.nist.gov/800-63-3/

[60] A. Narayanan and V. Shmatikov, "Fast dictionary attacks on passwords using time-space tradeoff," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, 2005, pp. 364–372.

[61] J. Goodman, "A bit of progress in language modeling," *Comput. Speech Language*, vol. 15, no. 4, pp. 403–434, 2001.

[62] J. Galbally, I. Coisel, and I. Sanchez, "A probabilistic framework for improved password strength metrics," in *Proc. IEEE Int. Carnahan Conf. Secur. Technol. (ICCST)*, May 2014, pp. 112–117.

[63] R. Veras, C. Collins, and J. Thorpe, "On semantic patterns of passwords and their security impact," in *Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, 2014, pp. 1–16.

[64] S. Houshmand and S. Aggarwal, "Building better passwords using probabilistic techniques," in *Proc. ACM Comput. Secur. Appl. Conf. (ACSAC)*, 2012, pp. 109–118.

[65] A. Ross, K. Nandakumar, and A. Jain, Eds., *Handbook of Multibiometrics*. New York, NY, USA: Springer, 2006.

[66] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Sov. Phys.-Dokl.*, vol. 10, no. 8, pp. 707–710, 1966.

[67] D. Jurafsky and J. H. Martin, *Speech and Language Processing*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1999.

[68] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, "Class-based n-Gram models of natural language," *J. Comput. Linguistics*, vol. 18, no. 4, pp. 467–479, 1992.

[69] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," Harvard Univ., Cambridge, MA, USA, Tech. Rep. TR-10-98, 1998.

[70] H. Tong, "Determination of the order of a Markov Chain by Akaike's information criterion," *J. Appl. Probab.*, vol. 12, no. 3, pp. 488–497, 1975.

[71] G. J. Lidstone, "Note on the general case of the Bayes–Laplace formula for inductive or *a posteriori* probabilities," *Trans. Faculty Actuaries*, vol. 8, pp. 182–192, Apr. 1920.

[72] F. Jelinek and R. L. Mercer, "Interpolated estimation of Markov source parameters from sparse data," in *Proc. Workshop Pattern Recognit. Practice*, 1980, pp. 381–397.

[73] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 226–239, Mar. 1998.

[74] A. Ross, *Handbook of Multibiometrics*. Springer, 2006, pp. 92–142.

[75] A. Jain, K. Nandakumar, and A. Ross, "Score normalization in multimodal biometric systems," *Pattern Recognit.*, vol. 38, no. 12, pp. 2270–2285, Dec. 2005.

[76] F. R. Hampel and P. J. Rousseeuw, *Robust Statistics: The Approach Basedon Influence Functions*. New York, NY, USA: Wiley, 1986.

[77] S. Ji, S. Yang, T. Wang, C. Liu, W.-H. Lee, and R. Beyah, "PARS: A uniform and open-source password analysis and research system," in *Proc. ACM Comput. Secur. Appl. Conf. (ACSAC)*, 2015, pp. 321–330.

[78] X. de Carne de Carnavalet and M. Mannan, "From very weak to very strong: Analyzing password-strength meters," in *Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, Feb. 2014, doi: http://dx.doi.org/ndss.2014.23268

[79] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 35, no. 3, pp. 400–401, Mar. 1987.

[80] S. Komanduri *et al.*, "Of passwords and people: Measuring the effect of password-composition policies," in *Proc. SIGCHI Conf. Human Factors Comput. Syst. (CHI)*, 2011, pp. 2595–2604.

Javier Galbally received the M.Sc. degree in electrical engineering from the Universidad de Cantabria, Santander, Spain, in 2005, and the Ph.D. degree in electrical engineering from the Universidad Autónoma de Madrid, Madrid, Spain, in 2009. In 2013, he joined the DG Joint Research Center, European Commission, where he is currently a Post-Doctoral Researcher. His research interests are mainly focused on pattern and biometric recognition, synthetic generation of biometric traits, and inverse biometrics, and started applying his knowledge in machine learning to password related problems.

Iwen Coisel received the Ph.D. degree in cryptography from the Orange Labs, Caen, France in 2009. The topic of the Ph.D. was anonymous authentication systems dedicated to low cost devices. He was with Orange Labs, Caen, France. He was a Researcher with the Crypto Group of the Université Catholique de Louvain, Belgium, where he was involved in private authentication systems. He has been a Scientific/Technical Project Officer with the Joint Research Center, European Commission, based in Ispra, Italy, since 2012.

Ignacio Sanchez received the M.Sc. degree in computer engineering from the University of Deusto, Bilbao, Spain, and the Ph.D. degree in computer engineering from the Spanish National University for Distance Education, Madrid, Spain. He has 14 years of experience in the domain of information security. He is currently with the Directorate General Joint Research Center, European Commission, where he is leading several research projects in the field of cybersecurity, privacy, and data protection.