PARS: A Uniform and Open-source Password Analysis and Research System

Shouling Ji[†], Shukun Yang[†], Ting Wang[‡], Changchang Liu[§], Wei-Han Lee[§], and Raheem Beyah[†]
[†]Georgia Institute of Technology, {sji, syang87}@gatech.edu, rbeyah@ece.gatech.edu
[‡]Lehigh University, ting@cse.lehigh.edu
[§]Princeton University, {cl12, weihanl}@princeton.edu

ABSTRACT

In this paper, we introduce an open-source and modular password analysis and research system, PARS, which provides a uniform, comprehensive and scalable research platform for password security. To the best of our knowledge, PARS is the first such system that enables researchers to conduct fair and comparable password security research. PARS contains 12 state-of-the-art cracking algorithms, 15 intra-site and cross-site password strength metrics, 8 academic password meters, and 15 of the 24 commercial password meters from the top-150 websites ranked by *Alexa*. Also, detailed taxonomies and large-scale evaluations of the PARS modules are presented in the paper.

Categories and Subject Descriptors

K.6.5 [**Security and Protection**]: Authentication

General Terms

Security

Keywords

Passwords, evaluation, cracking, measurement, metrics

1. INTRODUCTION

Text-based passwords have been the dominating means of computer system authentication for more than half a century. Although several shortcomings of passwords have been identified and extensively studied, passwords will likely remain as the primary computer authentication mechanism for the foreseeable future because of their significant advantages, e.g., high scalability, portability, and performance-price ratio, over their alternatives [1].

Over the past decade, password research has made considerable progress in many perspectives, e.g., password cracking, measurement, and security evaluation. On one hand, such extensive research on a multitude of topics has significantly enhanced people's comprehension of password security. On the other hand, this extensive focus on password security has made the status quo of password research disparate and somewhat chaotic, which causes confusion and makes it challenging to understand the current state of password security research. This is partly because much of the research has been conducted across different platforms with various settings which can make it challenging to accurately and fairly compare experimental results. For instance, which password cracking algorithm/tool is the most efficient in which scenario? How should password policies be created? How is passwords' strength accurately measured? How helpful and what is the effectiveness of password strength metrics and meters? What is the performance of commercial password meters?

Therefore, it is necessary and meaningful to conduct comparable research on a uniform and comprehensive platform where bias can be reduced and conclusions can be more persuasive. Aiming at this, in this paper, we make the following contributions.

(1) We propose and implement **PARS**, an *open-source* and modular **P**assword **A**nalysis and **R**esearch **S**ystem, to provide a *uniform and comprehensive research platform for password security.* (downloadable at [2]). To the best of our knowledge, PARS is the first such platform in the password research area that enables researchers to conduct fair and comparable password cracking, measurement, and evaluation studies.

(2) We systematically analyze and implement 12 state-of-the-art password cracking algorithms. Leveraging $\sim$ 115M real-world passwords, we evaluate the implemented algorithms in multiple scenarios. Based on our evaluation results, no cracking algorithm is optimal in all scenarios. The performance of an algorithm depends on multiple factors, e.g., algorithm design, target data, training data.

(3) We systematically analyze, implement, and evaluate 15 intra-site and cross-site password strength metrics, 8 academic password meters, and 15 of the 24 commercial password meters from the top-150 websites (ranked by *Alexa*, `http://www.alexa.com/`). According to our analysis and evaluation, most academic password metrics and meters are useful in helping system administrators estimate the security of passwords. However, for commercial password meters, their results are very inconsistent and their performance varies significantly. Moreover, some commercial meters are inaccurate and ineffective in guiding users to choose secure passwords. To make things worse, the feedback of

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
ACSAC '15, December 07 - 11, 2015, Los Angeles, CA, USA
ACM 978-1-4503-3682-6/15/12 ...$15.00.
http://dx.doi.org/10.1145/2818000.2818018.

some commercial meters may lead users to choose vulnerable passwords.

(4) We propose and implement a new metric, namely *Relative Improvement Ratio* (RIR), which quantitatively evaluates the relative performance of a password cracking algorithm over another in the same cracking setup. By the RIR analysis, PARS users can gain a more thorough understanding of the relationships between password cracking algorithms. With this analysis, RIR can also shed light on the need for designing *Hybrid Cracking Algorithms*.

2. PARS: SYSTEM OVERVIEW

In general, PARS has three functional modules.

Password Dataset Analysis. In this module, leveraging a large corpus ($\sim$ 115M) of real-world passwords, we develop several analytical functions to characterize password datasets by computing the distributions of passwords in terms of *length*, *structure*, and *composition*. The strength distribution of passwords in terms of specific metrics/meters can also be computed by interacting with the corresponding module in PARS. This module enables users/researchers to conduct statistical analysis of password datasets. Furthermore, to conduct uniform and comparable analysis, researchers can construct standard datasets from the original password datasets.

Password Crackability Evaluation. In this module, we implement state-of-the-art *password cracking algorithms*, including 7 academic password cracking algorithms proposed after 2005. Using this module, users can evaluate the crackability of password datasets under different scenarios, which enables them to comprehensively understand the vulnerability of a password dataset. Researchers can also use this module to uniformly and comparatively study the existing/newly developed password cracking algorithms.

Password Strength Evaluation. In this module, we implement 15 *strength metrics* and 8 *strength meters* from academia [3–11]. We also implement 15 of the 24 *commercial password meters* with online and offline versions, of the top-15 websites in 10 categories (*Business, Computers, Health, Science, Shopping, Society, News, Sports, Kids&Teens*, and *Home*) ranked by *Alexa*. Using this module, users can evaluate the strength of passwords in terms of any academic metric, meter, and/or commercial meter, which can help them understand the security of their passwords. Further, researchers can systematically study, evaluate and compare the existing/newly developed metrics/meters.

PARS, as a uniform and open-source platform, can help users gain meaningful insights on password security research. Using PARS, we introduce and develop the *Relative Improvement Ratio* (RIR), a metric that provides a theoretical quantification on the relative performance of one password cracking algorithm over another, which can shed light on the feasibility of designing hybrid password cracking algorithms.

In the following sections, we present and discuss the detailed implementation and evaluation of each module.

3. DATASETS AND ANALYSIS

Datasets. In this section, we present the 8 employed datasets as shown in Table 1 along with preliminary analysis. The datasets consist of about 115M real world passwords and cover several computer applications. They were leaked due to various password leakage incidents [3, 10, 12].

Table 1: Datasets.

	size	unique	language	website	type
`7k7k`	12.9M	3.5M	Chinese	7k7k.com/	game
`CSDN`	6.4M	4M	Chinese	csdn.net/	programmer
`Duduniu`	16.1M	10M	Chinese	duduniu.cn/	Internet cafe service
`Renren`	4.7M	2.8M	Chinese	renren.com/	social networks
`Tianya`	31M	12.6M	Chinese	tianya.cn/	Internet forum
`LinkedIn`	5.4M	4.9M	English	linkedin.com/	social networks
`Rockyou`	32.6M	14.3M	English	rockyou.com/	game
`Gamigo`	6.3M	6.3M	German	en.gamigo.com/	game

In Table 1, `7k7k`, `Rockyou`, and `Gamigo` are popular game websites; `CSDN` is a resource sharing website for programmers; `Duduniu` is a website of Internet cafe service software; `Renren` and `LinkedIn` are social networking services; and `Tianya` is an Internet forum. According to [3, 10, 12], most users of `7k7k`, `CSDN`, `Duduniu`, `Renren`, and `Tianya` are Chinese speaking users, most users of `LinkedIn` and `Rockyou` are English speaking users, and most users of `Gamigo` are German speaking users.

Ethical Discussion. Note that all the datasets in Table 1 are now publicly available. Further, these datasets have been extensively used for multi-purpose and meaningful academic research [3,10,12–15]. Although these real world passwords provide valuable resources to researchers, they were initially leaked illegally. Therefore, in this paper, we only use these data for research purposes.

Datasets Analysis. Let L, U, D, and S be four sets of characters to represent lower-case letters, upper-case letters, digits, and special symbols, respectively. A password is *Univariate* (Uni), *Bivariate* (Bi), *Trivariate* (Tri), or *Qualvariate* (Qual) if it is composed of one, two, three, or four set(s) of characters, respectively. Then, the password length, composition, and structure distributions (we focus on the top-10 popular password structures obtained in [10]) of the datasets in Table 1 are shown in Table 2. From Table 2, we observe that (*i*) the majority of passwords of all datasets have length less than 12. Particularly, a significant number of passwords of most datasets have length less than 8. However, `CSDN` has fewer passwords of length less than 8. This could be due to the fact that it introduced a new password policy of *minimum password length of* 8 while the website was live [10]. Thus, only a small amount of early users have passwords of length less than 8; (*ii*) the majority of passwords of most datasets are Uni and Bi. `LinkedIn` has more Tri and Qual passwords than other datasets, which suggests it has more stronger passwords with respect to composition complexity; and (*iii*) different datasets have very different structure distributions. Generally, $88.2\% - 97.8\%$ Chinese passwords follow the 10 most popular password structures. As for the English password datasets, 90% of `Rockyou`'s passwords follow the 10 most popular password structures. However, 22.9% of `LinkedIn`'s passwords do not follow the 10 most popular structures. An explanation could be that `LinkedIn` is a career-oriented professional social network service, and thus people pay more attention to their passwords. 41.8% of `Gamigo`'s passwords also do not have the 10 most popular structures. This is mainly because the popular structure conclusion in [10] is derived from analyzing Chinese and English passwords, which may not apply to the German password dataset `Gamigo`.

Standard Datasets. For our following evaluation and quantification, in order to *guarantee the fairness* and to *reduce possible bias caused by dataset size differences*, we ran-

Table 2: Password length, composition, and structure distributions.

	length distribution								composition distribution				structure distribution									
	≤6	7	8	9	10	11	12	≥13	Uni	Bi	Tri	Qual	LD	L	D	DL	LDL	UD	U	ULD	DLD	LDLD
7k7k	10.9%	21.6%	23.0%	17.9%	16.3%	6.5%	1.7%	2.1%	63.5%	35.1%	1.4%	0.0%	24.1%	9.9%	53.3%	5.1%	1.3%	0.5%	0.2%	0.2%	0.5%	0.3%
CSDN	1.7%	0.4%	26.0%	23.2%	17.6%	13.4%	7.5%	10.3%	44.9%	50.0%	4.8%	0.3%	33.4%	10.2%	34.1%	7.2%	2.2%	2.4%	0.6%	0.7%	0.7%	0.6%
Duduniu	7.9%	14.2%	20.6%	26.1%	18.8%	7.0%	2.4%	3%	37.0%	61.0%	2.0%	0.0%	45.9%	8.4%	28.3%	7.7%	1.9%	2.4%	0.3%	0.4%	0.5%	0.4%
Renren	17.8%	20.5%	21.8%	15.1%	10.2%	9.8%	2.1%	2.7%	62.8%	34.6%	2.5%	0.1%	24.6%	16.8%	45.3%	3.8%	1.6%	0.7%	0.7%	0.6%	0.4%	0.4%
Tianya	10.0%	15.2%	17.1%	13.1%	15.8%	10.0%	4.8%	13.9%	54.4%	36.3%	9.1%	0.1%	23.1%	8.5%	45.5%	6.1%	2.2%	0.9%	0.3%	0.5%	0.7%	0.5%
LinkedIn	11.5%	12.1%	32.1%	17.5%	12.5%	6.5%	3.9%	4%	26.1%	50.3%	19.8%	3.8%	30.1%	22.0%	3.5%	3.3%	6.7%	0.8%	0.6%	7.3%	1.2%	1.6%
Rockyou	15.6%	17.5%	20.7%	15.3%	14.0%	6.0%	3.8%	6.9%	44.2%	49.3%	6.1%	0.4%	33.1%	26.2%	16.4%	4.2%	2.7%	2.3%	1.6%	1.7%	0.8%	0.9%
Gamigo	7.0%	6.3%	20.9%	12.2%	44.4%	4.0%	2.9%	2.4%	17.5%	69.3%	13.1%	0.0%	21.9%	12.5%	4.7%	3.4%	10.3%	0.4%	0.3%	3.1%	0.9%	0.7%

domly and uniformly sample 2 million *unique* passwords as a *standard dataset* from each original dataset. Consequently, we obtain 8 standard datasets. In the rest of this paper, without specification, we use the standard datasets for the training in password crackability evaluation and other scenarios.

4. PASSWORD CRACKABILITY

4.1 Password Cracking Advances

In [16], Narayanan and Shmatikov proposed to use *standard Markov modeling techniques* to dramatically reduce the search size of password space. In [7], Castelluccia et al. improved the Markov model in [16]. They proposed to construct an *n-gram* based Markov model, under which to determine the probability of the next character, given the consideration of $n-1$ previous characters. However, the algorithm in [7, 16] cannot enumerate passwords in the *decreasing order of likelihood*. To address this limitation, Dürmuth et al. proposed an improved cracking algorithm, namely *Ordered Markov ENumerator* (OMEN) in [13], which can make password guesses in the decreasing order of likelihood. Furthermore, they also extended OMEN to OMEN+, where users' social profiles are considered in password cracking.

Taking another approach, Weir et al. proposed a password cracking algorithm using Probabilistic Context-Free Grammars (PCFGs) [17]. Basically, the cracking can be conducted in two steps. In the first step, PCFGs are generated based on a training dataset. In the second step, word-mangling rules are created from the trained PCFGs and password guesses are generated for actual cracking. From [8], the designed PCFG based cracking algorithm in [17] does not consider the probability of letter segments. To address this issue, Veras et al. in [14] proposed an improved PCFG based password cracking algorithm, where the grammars take into account *structure, syntax, and semantics* of passwords.

In [18], Zhang et al. studied the effect of expired passwords on the security of current passwords. Leveraging a dataset of 7.7K accounts, they proposed an approximately optimal searching strategy to crack users' new passwords from their expired passwords. Another similar scheme is presented in [12], where Das studied the password reuse problem. Based on 6K accounts, they designed a cracking algorithm to guess users' passwords on one site from their leaked passwords on other sites. Both the algorithms in [18] and [12] are based on some transformation and mangling rules, e.g., capitalization, insertion, deletion, and leet-speak.

There are also many password cracking tools available, among which the most popular one is John the Ripper (JtR) [19]. JtR supports multiple modes: *Wordlist mode* (JtR-W), *Single mode* (JtR-S), *Incremental mode* (JtR-I), and *Markov mode* (JtR-M). In JtR-W, a dictionary and a password hash file serve as inputs. JtR will try each word in the dictionary as a password guess to perform cracking. In JtR-S, each password hash serves as an input along with an auxiliary string, e.g., username. Then JtR-S applies a set of mangling rules to the auxiliary string to generate password guesses. JtR-I is an *intelligent brute force cracking method*, in which the *statistical character frequencies* will be employed to brute force the entire password space following a *quasi-decreasing order of password probabilities*. JtR-M is a similar Markov model based cracking strategy as in [16].

In [15], Amico et al. evaluated the Markov model and PCFG based cracking algorithms in [16] and [17], and JtR-M (all developed before 2010). According to their results, all the algorithms have their advantages in different scenarios. Different conclusions are drawn in [10], where Ma et al. evaluated *n*-gram Markov model and PCFG based password cracking schemes. They concluded that the Markov model along with different normalization and smoothing methods performs better than PCFG under the *probability threshold model* (we will discuss this model later). Consequently, inconsistent conclusions of existing password cracking algorithms remain. Furthermore, with the emergence of new cracking algorithms and new versions of cracking tools, e.g., OMEN [13], semantics based algorithm [14], JtR 1.7.9 [19], which cracking algorithm works best in what scenarios is not known. Therefore, a *uniform platform* to comprehensively and fairly evaluate existing cracking algorithms could be a tremendous resource to the password research community.

4.2 Implementation and Analysis

We implement all the password cracking algorithms summarized in Section 4.1. Besides that, we also integrate the popular password cracking tool John the Ripper (JtR) [19] into the crackability evaluation component. For convenience, we denote Narayanan and Shmatikov's algorithm as NS [16], Zhang et al.'s algorithm as ZMR [18], Veras et al.'s algorithm as VCT [14], and Das et al.'s algorithm as DBCBW [12] (initials of authors). Then, we summarize and analyze the implemented algorithms in Table 3, where Dic., Str., Syn., and Sem. stand for Dictionary, Structure, Syntax, and Semantics, respectively; SP indicates that an algorithm considers *Social Profiles* (SP) of users when cracking; DO indicates that an algorithm makes password guesses in the *Decreasing Order* (DO) of probabilities; CC indicates that the entire password space (i.e., the set of all the possible passwords) can be *Completely Covered* (CC) by an algorithm's guesses; On/Off indicates that an algorithm is an *Online/Offline* (On/Off) attack; G/S indicates that the algorithm is designed to perform a *General/Special* password attack; DT and BF indicate *Direct Try* and *Brute Force* (BF) respectively; and ●, ◑, and ○ represent *true*, *partially true*, and *false*, respectively.

Table 3: Password cracking algorithms analysis.

	Year	Dic.	Train	Model	Str.	Syn.	Sem.	SP	DO	CC	On/Off	G/S
NS	05	○	●	Markov	◑	◑	○	○	○	◑	○/●	G
PCFG	09	●	●	PCFG	●	○	○	○	●	◑	○/●	G
ZMR	10	○	◑	Heuristic	○	○	○	○	◑	○	●/●	S
OMEN	13	○	●	Markov	◑	◑	○	○	●	◑	○/●	G
OMEN+	13	○	●	Markov	◑	◑	○	●	●	◑	○/●	G/S
n-gram	12/14	○	●	Markov	◑	◑	○	○	○	◑	○/●	G
VCT	14	●	●	PCFG	●	●	●	○	●	◑	○/●	G
DBCBW	14	○	◑	Heuristic	○	○	○	○	◑	○	●/●	S
JtR-W	12	●	○	DT	○	○	○	○	○	○	○/●	G
JtR-S	12	○	○	Mangling	○	○	○	◑	○	○	○/●	S
JtR-I	12	○	○	BF	○	○	○	○	◑	●	○/●	G
JtR-M	12	○	●	Markov	◑	◑	○	○	●	◑	○/●	G

We discuss Table 3 as follows.

(1) JtR-W, PCFG, and VCT are dictionary-based attacks, while all the other attacks are dictionary-free. JtR-W, JtR-S, and JtR-I are training-free attacks. This is because that JtR-W cracks passwords by directly trying the words from a dictionary, JtR-S makes password guesses by applying mangling rules to an auxiliary string, and JtR-I is an intelligent brute force algorithm. All the Markov model and PCFG based algorithms are training-based since they need to build a model to generate password guesses in an optimal order. For the two heuristics based algorithms ZMR, which generates guesses by applying password transformation heuristics to expired passwords, and DBCBW, which generates guesses by applying password transformation heuristics to leaked passwords from other sites, they can be (partially) training-based or training-free. If the cracking heuristics are trained, they can be applied in an optimal order and thus accelerate the cracking process.

(2) All of the Markov model based algorithms partially consider passwords' structure and syntax. This is due to the property of the Markov model. If a password structure/syntax appears frequently in the training data, the *transition probability* of the corresponding structure/syntax path will be high in the Markov model. PCFG is a structure-based algorithm. During the process of password guesses generation, all the letter segments are replaced by words with the same length from an input dictionary. Therefore, PCFG does not consider password syntax or semantics. Because of similar reasons as discussed before, ZMR, DBCBW, JtR-W, JtR-S, and JtR-I do not employ password structure, syntax, or semantics information in their cracking. Note that VCT is the only algorithm that considers passwords' structure, syntax, and semantics, under which a PCFG model based password generator will be trained.

(3) OMEN+ is the only existing algorithm that uses users' social profiles to facilitate password cracking. The motivation of OMEN+ is that with a nontrivial probability, user-chosen passwords have correlations with their social profiles, e.g., name, birthday, and education. According to the results in [13], about 5% more passwords can be cracked compared to OMEN which does not use users' social profiles. Depending on the input auxiliary information (string), which could be the name, username, email, birthday, etc., JtR-S can be considered as a technique that partially considers users' social profiles.

(4) All the Markov model and PCFG based algorithms except NS and n-gram can generate password guesses in the decreasing order of likelihood, which is intuitively the optimal strategy for password cracking. ZMR and DBCBW can generate password guesses in a quasi-decreasing order of probabilities depending on whether their password transformation heuristics are trained. Also, by employing the *statistical character frequency information*, JtR-I can make password guesses in a quasi-decreasing order of probability.

(5) According to our previous discussion, it is intuitive that ZMR, DBCBW, JtR-W, and JtR-S cannot completely cover the entire password space. It is also intuitive that JtR-I can completely cover the entire password space since it is a brute force algorithm. For all the other algorithms, it is possible however without guarantee that they can completely cover the password space. For Markov model based algorithms, it depends on whether the trained Markov model can cover the password space, and for PCFG based algorithms, it depends on the trained PCFGs and the used dictionaries. Generally speaking, given the same training data, NS [16] is more likely to cover the entire password space.

(6) Other than ZMR and DBCBW, all the other algorithms are initially designed for an offline attack. ZMR and DBCBW can be used for both online and offline attacks. This is because in ZMR and DBCBW, an adversary is assumed to have a victim's expired password information and password information from other sites respectively, and thus an adversary can achieve a high probability of success with a small number of guesses based on the auxiliary knowledge.

(7) As we discussed before, ZMR is an expired password based attack, DBCBW is a password reuse based attack, and JtR-S is an auxiliary information based attack for specified users. Therefore, these three algorithms can be employed to perform special purpose attacks. OMEN+ can be used for both special and general purpose attacks, depending on the available social profiles of victims. All the other attacks are designed to perform general purpose password cracking.

4.3 Evaluation

In this subsection, we evaluate our implementation. Since we do not have any available expired passwords, reused passwords, or the social profiles of any users, we do not evaluate ZMR, DBCBW, and OMEN+ at this moment[1].

Evaluation Setting. In the evaluation, JtR-W uses its default dictionary. For other algorithms that need an input dictionary, we use the combination of the widely employed *Dic-029* [17] and *Pinyin* [3]. When evaluating n-gram [7] [10], we use 3-gram (3g). For JtR-W, its guesses are the input dictionary words. For other algorithms, they will make guesses based on their models. Here, we limit each algorithm (except for JtR-W) to generate two billion guesses[2]. Furthermore, to make the comparison fair and reduce possible bias, all the training and testing data are the standard datasets.

Results and Analysis. The results of training-free and training-based cracking algorithms are shown in Tables 4 and 5, respectively. We analyze the results as follows.

[1]We could crawl the web for the social profiles of victims who have emails available in Table 1. However, this raises ethical concerns and could be illegal.

[2]It is straightforward for us to generate more guesses or even exhaust the entire password space. However, since our main purpose is to implement existing password cracking algorithms in a uniform research system, we here only show their performance on this platform. Further, it takes significant more time for some algorithms to generate a large number guesses. For instance, it takes VCT several days to generate two billion guesses on a moderate PC.

Table 5: Training-based password cracking. Each value indicates the percentages of passwords been cracked.

Training	7k7k						CSDN						Duduniu						Renren					
Data	PCFG	VCT	NS	3g	OMEN	JtR-M	PCFG	VCT	NS	3g	OMEN	JtR-M	PCFG	VCT	NS	3g	OMEN	JtR-M	PCFG	VCT	NS	3g	OMEN	JtR-M
7k7k	–	–	–	–	–	–	4.9	6.2	0.3	1.9	20.6	22.6	5.3	6.9	0.5	8.1	21.1	23.3	13.2	14.6	1.7	25.0	48.5	51.5
CSDN	9.4	11.8	0.7	3.6	30.9	69.1	–	–	–	–	–	–	12.4	23.8	0.3	1.8	19.0	**36.4**	11.8	17.5	0.9	5.1	24.0	57.2
Duduniu	10.0	12.3	1.2	10.2	68.7	69.5	10.8	19.4	0.2	1.4	**27.6**	26.8	–	–	–	–	–	–	23.4	30.0	1.4	12.7	52.4	58.1
Renren	13.1	15.0	1.7	15.9	71.3	**73.8**	10.9	15.0	0.3	1.7	25.3	24.3	24.7	32.6	0.6	5.9	29.4	30.0	–	–	–	–	–	–
Tianya	11.5	14.2	1.5	12.9	71.1	72.3	11.5	20.1	0.3	1.7	26.7	26.4	15.4	25.3	0.6	5.3	29.5	35.6	14.7	19.9	1.9	15.6	53.1	**59.3**
LinkedIn	2.4	3.5	0.0	2.9	36.1	50.6	6.6	6.0	0.0	0.4	15.4	11.5	11.6	12.3	0.1	1.8	14.5	20.9	10.0	15.5	0.3	5.0	32.8	46.8
Rockyou	3.2	3.8	0.1	7.5	62.8	62.2	6.8	6.0	0.1	1.2	20.2	18.7	12.2	12.9	0.2	4.3	22.7	26.0	11.2	15.9	0.5	11.7	50.6	53.9
Gamigo	1.4	2.8	0.0	2.1	8.9	37.4	6.0	5.1	0.0	0.4	4.9	5.2	10.2	10.3	0.0	1.4	4.4	16.1	8.5	13.1	0.1	3.7	10.3	37.3

Training	Tianya						LinkedIn						Rockyou						Gamigo					
Data	PCFG	VCT	NS	3g	OMEN	JtR-M	PCFG	VCT	NS	3g	OMEN	JtR-M	PCFG	VCT	NS	3g	OMEN	JtR-M	PCFG	VCT	NS	3g	OMEN	JtR-M
7k7k	8.5	9.8	0.7	15.3	33.9	36.1	2.5	1.9	0.3	2.9	3.6	4.3	2.8	2.5	0.5	6.2	9.2	10.7	1.4	1.3	0.2	2.1	3.6	4.2
CSDN	8.8	17.1	0.3	2.2	19.9	**42.6**	9.5	10.6	0.1	0.8	2.5	14.4	8.7	12.3	0.2	1.8	5.4	26.2	7.4	7.5	0.1	0.9	2.7	10.7
Duduniu	10.3	17.3	0.6	6.6	38.2	42.2	15.5	18.9	0.3	3.4	5.7	15.5	15.7	22.8	0.4	5.7	12.4	26.5	10.6	12.5	0.1	2.6	5.2	11.3
Renren	11.4	15.9	1.0	9.7	37.5	39.9	17.5	21.9	0.4	5.7	9.4	13.7	17.5	25.2	0.5	8.2	16.2	22.1	11.5	13.7	0.3	3.6	7.0	9.3
Tianya	–	–	–	–	–	–	14.0	14.1	0.3	3.9	5.1	14.0	14.1	16.9	0.5	6.8	12.8	25.9	9.6	9.1	0.2	3.0	4.9	10.5
LinkedIn	6.6	7.9	0.2	2.1	21.0	29.7	–	–	–	–	–	–	16.4	**31.3**	0.1	4.0	18.8	30.2	11.2	17.3	0.0	2.1	11.1	14.1
Rockyou	7.1	8.4	0.3	5.6	34.6	36.7	17.2	**27.5**	0.1	5.9	17.0	22.5	–	–	–	–	–	–	11.5	**17.9**	0.1	4.4	12.9	15.7
Gamigo	5.5	6.5	0.2	1.7	5.4	22.5	15.3	23.4	0.0	1.4	6.0	19.1	14.9	27.4	0.1	2.5	6.9	26.8	–	–	–	–	–	–

Table 4: Training-free password cracking. Each value indicates the percentages of passwords been cracked.

	7k7k	CSDN	Duduniu	Renren	Tianya	LinkedIn	Rockyou	Gamigo
JtR-W	0.0	0.4	0.9	1.6	0.5	1.7	1.1	1.1
JtR-I	23.7	6.3	11.7	28.1	15.5	17.0	24.9	11.2

(1) By trying a dictionary consisting of frequently used words (passwords), JtR-W can crack 1.7% and 1.6% passwords of LinkedIn and Renren, respectively. This implies that some users of these two datasets just choose some common passwords. More impressively, without any training knowledge, JtR-I can crack $\sim 1/4$ passwords of Renren, Rockyou, and 7k7k. This is because these three datasets have more passwords that are short, simply composed, and with popular structures.

(2) From Table 5, we can see that when cracking 7k7k, Renren-trained JtR-M has the best performance; when cracking CSDN, Duduniu-trained OMEN has the best performance; and when cracking Rockyou, LinkedIn-trained VCT has the best performance. Therefore, *no cracking algorithm is always optimal. The actual cracking performance depends on multiple factors, e.g., training data, targeting data, and cracking algorithm.*

(3) *Following the above observation, when cracking a password/dataset, it is important to choose both proper training data and a proper cracking algorithm.* For instance, Duduniu-trained OMEN has much better performance to crack CSDN than Gamigo-trained OMEN. This is because the two Chinese datasets Duduniu and CSDN are more structurally, syntactically, and semantically similar than that of Gamigo and CSDN. In addition, Rockyou-trained VCT has better performance than Rockyou-trained NS when cracking LinkedIn. This is because although Rockyou and LinkedIn are structurally, syntactically, and semantically similar, VCT considers password structure, syntax, and semantics during the training phase while NS only partially considers password structure and syntax.

(4) In most of the cases, VCT has better performance than PCFG. This is because during the training phase, VCT considers password structure, syntax, and semantics information together while PCFG only considers password structure information (see Table 3). However, in the scenarios that the training and targeting datasets are syntactically and semantically different (although this does not happen frequently), PCFG may perform better than VCT.

(5) For the Markov model based algorithms, OMEN and JtR-M have better performance than NS and 3g. The main reason is that OMEN and JtR-M generate password guesses in the decreasing order of likelihood (the optimal password cracking strategy), while NS and 3g generate password guesses above some predefined threshold (not necessarily following the decreasing order of likelihood). Furthermore, 3g performs better than NS. This is because NS is actually a 1-gram Markov model based algorithm, which implies that 3g considers more information when training the password cracking model. Hence, 3g can generate more accurate guesses than NS. However, as we analyzed before, among all the Markov model based algorithms, NS has the largest guess space, i.e., NS has the largest password space coverability.

(6) Gamigo only has at most 17.9% passwords cracked among all the evaluation scenarios. The reason is that all the other datasets are from different lingual/cultural domains other than Gamigo. Therefore, the cracking algorithms trained by these datasets are not effective when making guesses. This can be confirmed by the results in Table 2. Gamigo is not structurally, syntactically, or semantically similar to other datasets. Similarly, LinkedIn has at most 27.5% passwords cracked since it has more passwords with unpopular password structures than other Chinese and English datasets.

5. PASSWORD MEASUREMENT

5.1 Password Measurement Advances

Leveraging the single-sign-on passwords used by 25K faculty, staff, and students at CMU, Mazurek et al. measured the password guessability of university passwords [20]. Another interesting work is [21], where based on a survey of 470 CMU computer users, Shay et al. analyzed users' attitudes and behaviors when encountering stronger password requirements. In [3], Li et al. conducted an empirical analysis of Chinese web passwords. According to their statistical results, user-chosen passwords have explicit regional differences. In [4], Bonneau analyzed an anonymized corpus of 70M Yahoo! passwords.

In [8], Weir et al. evaluated testing metrics, e.g., NIST entropy, for password creation policies by attacking revealed passwords using their PCFG based cracking algorithm. They found that the NIST entropy is not an effective metric for

password security, and proposed new PCFG cracking based password creation policies. Another work employing the password cracking idea to measure password strength is [9], where Kelley analyzed 12K passwords collected under seven composition policies via an online study. In [5], Houshmand and Aggarwal proposed a tool, named *Analyzer and Modifier for Passwords* (AMP), to help users choose stronger passwords. AMP first estimates a password's crackability based on the PCFG cracking model, and then modifies the weak password slightly to meet the security requirement. Komanduri et al. implemented another tool, namely *Telepathwords*, to help users create strong passwords [22]. In [23], Forget et al. also developed a tool, namely *Persuasive Text Passwords* (PTP), which leverages the persuasive technology principle to influence users in creating more secure passwords.

In [6], Ur et al. studied the effect of strength meters on password creation. They found that significant increases in password resistance were only achieved using meters that scored passwords stringently. Another work studying existing password meters is [24], where Carnavalet and Mannan analyzed 11 commercial meters. They found significant inconsistency among different meters. To improve the accuracy of password strength measurement, Castelluccia et al. presented *adaptive password strength meters* that estimate password strength using Markov models [7]. They also proposed a secure implementation of the presented concept. In [10], Ma et al. proposed a *probability-threshold graph model* to capture the probability threshold distribution in log scale versus the percentage of passwords above the threshold.

5.2 Strength Metrics and Analysis

Metrics. Before giving the formal definitions of strength metrics, we first provide some preliminary definitions. Let U be the *universal set of all the possible passwords*. Then, mathematically, any password dataset (e.g., `LinkedIn`) can be defined as a subset, denoted by S, of U with a specific probability distribution $P = \{p_i | i \in S$, i.e., i is a password in $S\}$. Let $m = |S|$ and without loss of any generality, within $P(S)$, we assume $p_1 \geq p_2 \geq \cdots \geq p_m$.

In PARS, we implement 15 mathematical strength metrics of password cracking difficulty, of which 12 are developed/used by Bonneau in [4] and Li et al. in [3]. In addition, we give 3 new metrics for cross-site password cracking by extending Bonneau and Li et al.'s definitions. Basically, all the 15 metrics can be partitioned into two categories: *intra-site metrics*, which measure the cracking difficulty of a password dataset directly, and *cross-site metrics*, which measure the cracking difficulty of a password dataset given another dataset. First, given S and P, we present the intra-site metrics as follows.

(1) *Min-entropy* H_∞. Min-entropy is defined as $H_\infty = -\log p_1$, which denotes the *worst-case security metric* against an attacker.

(2) *Guesswork* $G(P)$ *and* $\widetilde{G}(P)$. Guesswork is defined as the expected number of guesses to find the password of an account in the *optimal guessing order* (password probability decreasing order), and formally, $G(P) = \sum_{i=1}^{m} i \cdot p_i$. $\widetilde{G}(P)$ is the bit/entropy-form of $G(P)$ and defined as $\widetilde{G}(P) = \log(2G(P) - 1)$.

(3) *β-success-rate* $\lambda_\beta(P)$ *and* $\widetilde{\lambda}_\beta(P)$. $\lambda_\beta(P)$ measures the expected success probability to find the password of an account given β guesses. Formally, $\lambda_\beta(P) = \sum_{i=1}^{\beta} p_i$. $\widetilde{\lambda}_\beta(P)$ is $\lambda_\beta(P)$'s bit-form and $\widetilde{\lambda}_\beta(P) = \log(\frac{\beta}{\lambda_\beta(P)})$.

(4) *α-work-factor* $\mu_\alpha(P)$ *and* $\widetilde{\mu}_\alpha(P)$. $\mu_\alpha(P)$ measures the expected number of guesses needed to crack α proportion of accounts. Formally, $\mu_\alpha(P) = \min\{j | \sum_{i=1}^{j} p_i \geq \alpha\}$. $\widetilde{\mu}_\alpha(P)$ is $\mu_\alpha(P)$'s bit-form, and $\widetilde{\mu}_\alpha(P) = \log(\frac{\mu_\alpha(P)}{\lambda_{\mu_\alpha}})$.

(5) *α-guesswork* $G_\alpha(P)$ *and* $\widetilde{G}_\alpha(P)$. $G_\alpha(P)$ measures the expected number of guesses per account to achieve a success probability of α. Formally, $G_\alpha(P) = (1 - \lambda_{\mu_\alpha})\mu_\alpha + \sum_{i=1}^{\mu_\alpha} i \cdot p_i$. $\widetilde{G}_\alpha(P)$ is $G_\alpha(P)$'s bit-form and $\widetilde{G}_\alpha(P) = \log(\frac{2G_\alpha(P)}{\lambda_{\mu_\alpha}} - 1) + \log\frac{1}{2-\lambda_{\mu_\alpha}}$.

Now, suppose we have another password dataset $V \subseteq U$ with probability distribution Q. Given S, to measure the cracking difficulty of V, Li et al. extended $\lambda_\beta(P)$, $\mu_\alpha(P)$, and $\widetilde{\mu}_\alpha(P)$ to the cross-site cracking scenario [3]. Similarly, we further extend $\widetilde{\lambda}_\beta(P)$, $G_\alpha(P)$, and $\widetilde{G}_\alpha(P)$ to the cross-site scenario. Given $p_i \in P$, let $f(p_i) = q_j$ such that i and j denote the same password in U and $q_j \in Q$ is the probability of $j \in V$, i.e., p_i and q_j are the probabilities of the same password within two different datasets. Then, the cross-site metrics are formally defined as follows (we use the superscript c to denote the cross-site scenario).

(1) *Cross-site β-success-rate* $\lambda^c_\beta(P)$ *and* $\widetilde{\lambda}^c_\beta(P)$. $\lambda^c_\beta(P) = \sum_{i=1}^{\beta} f(p_i)$, and $\widetilde{\lambda}^c_\beta(P) = \log(\frac{\beta}{\lambda^c_\beta(P)})$.

(2) *Cross-site α-work-factor* $\mu^c_\alpha(P)$ *and* $\widetilde{\mu}^c_\alpha(P)$. $\mu^c_\alpha(P) = \min\{j | \sum_{i=1}^{j} f(p_i) \geq \alpha\}$, and $\widetilde{\mu}^c_\alpha(P) = \log(\frac{\mu^c_\alpha(P)}{\lambda^c_{\mu^c_\alpha}})$.

(3) *Cross-site α-guesswork* $G^c_\alpha(P)$ *and* $\widetilde{G}^c_\alpha(P)$. $G^c_\alpha(P) = (1 - \lambda^c_{\mu^c_\alpha})\mu^c_\alpha + \sum_{i=1}^{\mu^c_\alpha} i \cdot f(p_i)$, and $\widetilde{G}^c_\alpha(P) = \log(\frac{2G^c_\alpha(P)}{\lambda^c_{\mu^c_\alpha}} - 1) + \log\frac{1}{2-\lambda^c_{\mu^c_\alpha}}$.

Table 6: Intra-site metrics evaluation.

	H_∞	$\widetilde{G}$	$\widetilde{\lambda}_5$	$\widetilde{\lambda}_{10}$	$\widetilde{\mu}_{.25}$	$\widetilde{\mu}_{.5}$	$\widetilde{G}_{.25}$	$\widetilde{G}_{.5}$
7k7k	4.78	20.92	6.29	7.09	15.63	19.85	15.50	19.50
CSDN	4.77	21.29	5.73	6.58	15.76	20.65	15.60	20.29
Duduniu	6.12	22.62	7.61	8.26	19.24	21.91	19.10	21.62
Renren	4.74	20.72	6.40	7.12	16.21	19.81	16.07	19.47
Tianya	4.70	22.54	6.30	7.11	15.25	20.08	15.11	19.68
LinkedIn	15.82	22.10	16.23	16.76	21.72	22.08	21.69	22.01
Rockyou	6.81	22.64	8.19	8.93	15.97	20.14	15.84	19.76
Gamigo	21.59	22.59	21.59	21.82	22.59	22.59	22.59	22.59

Metrics Evaluation. Now, we evaluate the password guessing difficulty of passwords (Table 1) with respect to the implemented intra-site and cross-site metrics as shown in Tables 6 and 7, respectively. From Table 6, we have the following observations.

(1) The H_∞ of most datasets is very low, e.g., H_∞(`Tianya`) = 4.7, which implies the most popular passwords of these datasets are easily guessable. `Gamigo` and `LinkedIn` have larger H_∞, which implies the most popular passwords within these two datasets are more secure than those of other datasets. From Table 6, we see that $\widetilde{G}$ is not closely related to the crackability of a dataset.

Table 7: Cross-site metrics evaluation.

Given	7k7k			CSDN			Duduniu			Renren			Tianya			LinkedIn			Rockyou			Gamigo		
Dataset	$\widetilde{\lambda}^c_{10}$	$\widetilde{\mu}^c_{.05}$	$\widetilde{G}^c_{.05}$	$\widetilde{\lambda}^c_{10}$	$\widetilde{\mu}^c_{.05}$	$\widetilde{G}^c_{.05}$	$\widetilde{\lambda}^c_{10}$	$\widetilde{\mu}^c_{.05}$	$\widetilde{G}^c_{.05}$	$\widetilde{\lambda}^c_{10}$	$\widetilde{\mu}^c_{.05}$	$\widetilde{G}^c_{.05}$	$\widetilde{\lambda}^c_{10}$	$\widetilde{\mu}^c_{.05}$	$\widetilde{G}^c_{.05}$	$\widetilde{\lambda}^c_{10}$	$\widetilde{\mu}^c_{.05}$	$\widetilde{G}^c_{.05}$	$\widetilde{\lambda}^c_{10}$	$\widetilde{\mu}^c_{.05}$	$\widetilde{G}^c_{.05}$	$\widetilde{\lambda}^c_{10}$	$\widetilde{\mu}^c_{.05}$	$\widetilde{G}^c_{.05}$
7k7k	–	–	–	9.89	9.69	9.71	7.32	6.24	6.23	7.11	5.85	5.84	7.27	6.27	6.25	20.2	25.22	25.2	7.77	8.55	8.53	∞	26	26
CSDN	7.15	7.00	7.02	–	–	–	8.07	7.53	7.54	7.15	6.83	6.84	7.15	6.83	6.85	19.8	25.6	25.6	7.14	6.98	6.99	∞	26.2	26.2
Duduniu	8.58	11.4	11.4	11	15.4	15.4	–	–	–	8.50	11.4	11.4	8.59	11.8	11.8	18.1	25.4	25.4	8.88	14.9	14.8	∞	26.1	26.1
Renren	7.19	6.16	6.15	9.75	9.64	9.66	7.37	6.31	6.30	–	–	–	7.35	6.55	6.53	17	24.6	24.6	7.57	7.43	7.40	∞	25.3	25.3
Tianya	7.28	6.19	6.17	9.82	9.60	9.62	7.23	6.19	6.18	7.27	5.80	5.78	–	–	–	17.9	25.2	25.2	7.68	8.49	8.46	∞	26	26
LinkedIn	22	34.9	34.9	22.7	35.5	35.5	22.9	34.3	34.2	22	33.6	33.6	22.7	35.3	35.2	–	–	–	21.6	23.7	23.7	∞	26.9	26.9
Rockyou	9.61	17	17	11.5	18.6	18.6	9.38	14.2	14.2	9.36	13.1	13.1	9.54	16.8	16.7	14.7	23.8	23.8	–	–	–	∞	24.2	24.2
Gamigo	25.9	35.5	35.5	24.3	36.1	36.1	24.9	34.6	34.6	25.9	34.4	34.4	25.9	35.5	35.5	25.9	33.8	33.7	25.9	24.8	24.8	–	–	–

Table 8: Academic password strength meters.

	Year	Methodology	Model	Training	Outcome
NIST	06	rule-based			entropy
CMU	10	rule-based		✔	entropy
PCFG	10/12	attack	PCFG	✔	probability
Ideal	12	statistics			entropy
Adaptive	12	attack	Markov	✔	entropy
BFM	12	attack	Markov	✔	guess number
Weir	12	attack	PCFG	✔	guess number
PTG	14	attack	Markov/PCFG	✔	prob. threshold

(2) Gamigo and LinkedIn also have larger $\widetilde{\lambda}_\beta$, $\widetilde{\mu}_\alpha$, and $\widetilde{G}_\alpha$ values than other datasets. This implies they are less crackable than other datasets, which is generally consistent with the results in Table 5. From this point of view, the defined strength metrics are helpful in understanding the security of a password dataset.

From Table 7, we make the following observations.

(1) For most of the Chinese datasets, they have smaller $\widetilde{\lambda}^c_\beta$, $\widetilde{\mu}^c_\alpha$, and $\widetilde{G}^c_\alpha$ values given other Chinese datasets than given English/German datasets. This implies Chinese datasets have more similar password distributions. It follows that when cracking a Chinese dataset, an algorithm trained by another Chinese dataset could be more effective, which agrees with the results in Table 5 and the observations in [3]. A similar observation is also applicable to LinkedIn. Rockyou still has larger $\widetilde{\lambda}^c_\beta$, $\widetilde{\mu}^c_\alpha$, and $\widetilde{G}^c_\alpha$ values given LinkedIn. This is caused by the password distribution difference of Rockyou and LinkedIn as shown in Table 2.

(2) As a German dataset, Gamigo has a different password distribution than Chinese/English datasets. Therefore, Gamigo has larger $\widetilde{\lambda}^c_\beta$, $\widetilde{\mu}^c_\alpha$, and $\widetilde{G}^c_\alpha$ values in all the evaluation scenarios. This is consistent with the results in Table 5. Furthermore, since LinkedIn also has an apparent dissimilar password distribution as shown in Table 2, LinkedIn has high $\widetilde{\lambda}^c_\beta$, $\widetilde{\mu}^c_\alpha$, and $\widetilde{G}^c_\alpha$ values given other datasets. Because of the same reason, given LinkedIn, other datasets also have high $\widetilde{\lambda}^c_\beta$, $\widetilde{\mu}^c_\alpha$, and $\widetilde{G}^c_\alpha$ values.

In summary, most of the evaluation results of the intra-site and cross-site metrics are generally consistent with the cracking results in Table 5. Therefore, these metrics are helpful to estimate the security of a password dataset, given the availability of the password distribution information.

5.3 Academic Strength Meters and Analysis

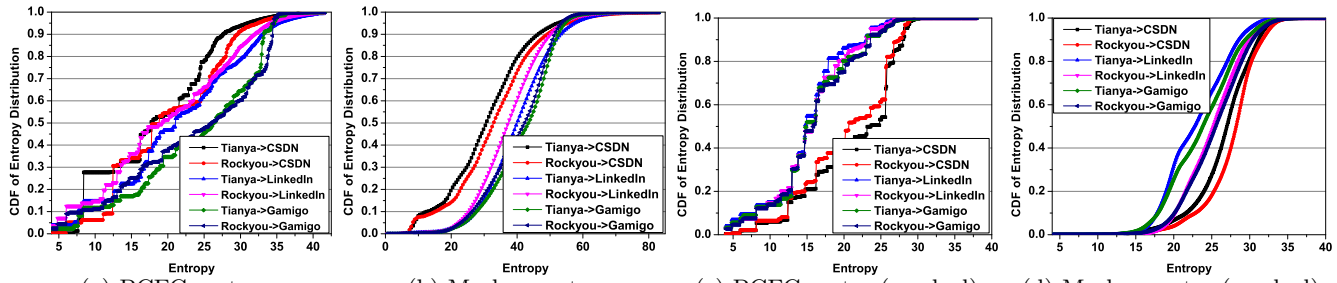
Academic Meters. In PARS, we implement and evaluate 8 popular strength meter proposals from academia: *NIST meter* [11], *CMU meter* [21], *PCFG meter* [8] [5], *Ideal meter* [7], *Adaptive meter* [7], *Brute Force Markov (BFM) meter* [9], *Weir meter* [9], and *Probability-Threshold Graph (PTG)* [10]. Below, we summarize and analyze the 8 meters in Table 8.

(1) NIST and CMU are *rule-based* meters. Specifically, NIST assigns an entropy score to a password based on certain rules, e.g., *the entropy of the first character is 4 bits, the entropy of the next 7 characters are 2 bits per character.* CMU assigns an entropy score to a password based on statistical rules trained from some password dataset. The methodologies of other academia meters are either *statistics-based* or *attack-based*. For the statistics-based Ideal meter, it assigns an entropy score to a password of a dataset based on the statistics on that dataset, e.g., *probability distribution.* For attack-based meters, they measure the strength of passwords by estimating the difficulty of using a cracking model to attack these passwords.

(2) The attack-based meters leverage either PCFG-based cracking algorithms [14, 17] or Markov-based cracking algorithms [13, 16]. For PTG, although it is Markov-based in [10], actually, it can also be implemented by leveraging other cracking algorithms, e.g., PCFG-based algorithms [17]. To build these meters, a pre-training phase is necessary. Therefore, the performance of all the attack-based meters is cracking algorithm dependent and training data dependent.

(3) The strength outcomes of different meters varies. NIST, CMU, Ideal, and Adaptive employ entropy; PCFG computes the cracking probability of a password using the PCFG-based cracking algorithm in [17]; BFM and Weir plot the number of cracked passwords given a number of guesses; and PTG plots the probability threshold versus the percentage of passwords above the threshold.

Academic Meters Evaluation. Here, we evaluate the performance of academic meters. Due to the space limitation, we put the evaluation results of NIST, CMU, and Ideal in [2]. Basically, according to our evaluation, the results of the NIST meter have little correlation with the actual password security, especially in the training-based cracking scenarios, which is consistent with observations in many existing works, e.g., [8]. CMU is another rule-based meter similar to NIST. The actual performance of the CMU meter depends on the training dataset. The Ideal meter is only a theoretical meter [7]. Its results come directly from the actual password distribution of a dataset. For PCFG-based meters PCFG [5, 8] and Weir [9], although they have different types of outcomes (e.g., probability, guess number), their performance is equivalent with respect to measuring password strength distribution. The reason is that they rely on the same cracking algorithm PCFG [17]. Similarly, for Markov-based meters [7] [9] [10], their performance is also equivalent on measuring password strength distribution since they are all based on the same cracking model.



(a) PCFG meter (b) Markov meter (c) PCFG meter (cracked) (d) Markov meter (cracked)

Figure 1: Academic meters evaluation.

Therefore, we use PCFG and Markov to represent PCFG-based and Markov-based meters, respectively. To make the evaluation results comparable, we convert all the meters' outputs to the entropy distribution. Furthermore, we employ `Tianya` and `Rockyou` as example training datasets and `CSDN`, `LinkedIn`, and `Gamigo` as example testing datasets for evaluating meters (complete evaluation scenarios of the 8 datasets are available at [2]). The results are shown in Fig.1, where (a) and (b) show the entropy distribution of all the passwords in `CSDN`, `LinkedIn`, and `Gamigo`, and (c) and (d) show the entropy distribution of the cracked passwords of `CSDN`, `LinkedIn`, and `Gamigo` by `Tianya` and `Rockyou` trained PCFG and OMEN. From Fig.1, we have the following observations.

(1) From Fig.1 (a) and (b), we see that most passwords have their entropy in range [10, 35] under the PCFG meter and [20, 60] under the Markov meter. Both meters assign more entropy to `Gamigo` than `CSDN` and `LinkedIn`. This implies `Gamigo` is more secure than `CSDN` and `LinkedIn`. Further, `LinkedIn` has higher entropy than `CSDN`, and thus `CSDN` is more crackable than `LinkedIn`. Generally, the above observations are consistent with our results in Table 5.

(2) From Fig.1 (a) and (b), we can also see that for `CSDN`, both `Tianya`-trained meters assign lower entropy than both `Rockyou`-trained meters. This implies `CSDN` is more vulnerable given `Tianya` than given `Rockyou`. On the other hand, according to the results of the two meters, `LinkedIn` is more vulnerable given `Rockyou` than given `Tianya`. These two observations are also consistent with our results in Table 5. For `Gamigo` under the PCFG meter, its low entropy (weak) passwords are more vulnerable given `Rockyou` while its high entropy (strong) passwords are more vulnerable given `Tianya` according to the results. For `Gamigo` under the Markov meter, it is more vulnerable given `Rockyou`, which is generally consistent with the results in Table 5.

(3) From Fig.1 (c) and (d), we see that most of the cracked passwords have entropy in the range of [5, 30] under the PCFG meter and [15, 35] under the Markov meter. Comparing the results with those in Fig.1 (a) and (b), we conclude that most of the crackable passwords have low entropy.Therefore, both the PCFG meters [5,8,9] and Markov meters [7, 9] [10] are meaningful in practice. Furthermore, the cracked passwords of `LinkedIn` and `Gamigo` have lower entropy than that of `CSDN`. Again, this implies `LinkedIn` and `Gamigo` are more secure than `CSDN`.

5.4 Commercial Meters and Analysis

Commercial Meters. Besides academic metrics and meters, we also implement 15 commercial password strength meters. First, we examine the top-15 sites in 10 categories ranked by *Alexa*. We summarize the basic statistics of the

Table 9: Summarization of top-150 commercial password policy checkers/meters.

Category	length		composition						meter	No Policy	N/A
	LL	UL	C1	C2	C3	C4	C5	C6			
Business	13	9		5	1	2	1	2	2		2
Computers	14	1			1		1	1	5	1	
Health	7	1	1			2			2		8
Science	8	2				1			3	2	5
Shopping	13	6				2	1	2	5		
Society	3	1							1	2	10
News	9	4		1					2		5
Sports	12	2				1			1		3
Kids&Teens	12	6	1			4			2	1	1
Home	12	5		1		2			1	1	1
Total	**103**	**37**	**2**	**7**	**2**	**14**	**3**	**5**	**24**	**7**	**35**

150 websites in Table 9 (detailed information and statistics are available at [2]), where *Lower Limit* (LL) and *Upper Limit* (UL) denote the minimum and maximum length requirements on acceptable passwords respectively, C1 (*letters only*), C2 (*letters or/and digits only*), C3 (*letters, digits or/and symbols*), C4 (*letters or/and digits only, at least one digit*), C5 (*letters, digits, or/and symbols, at least one digit/symbol*), and C6 (*Trivariate*) are six password composition policies, and N/A implies that either a site does not support the user registration function or the registration phase is not accessible unless the user is a customer/member of that site (e.g., Bank of America). Furthermore, the values in Table 9 indicate the number of sites meeting the length/composition policy requirement, having password meters, having no policy requirement, or N/A. From Table 9, we have the following observations.

(1) 103 sites have minimum password length constraints. Further, the sites that have many high-value accounts, e.g., Computers/Shopping/Business sites, tend to require users to choose longer passwords. On the other hand, 37 sites have explicit maximum password length constraints. Actually, many sites do have constraints on maximum password length although they do not explicitly state that.

(2) Among the 150 sites, 33 have explicit password composition policies (C1-C6). 7 sites have no policy at all. 24 sites have password meters that can assign a score/label for an input password. One third of the Shopping/Computers sites employ meters to indicate to users the password strength.

Based on the techniques in [24], we implement both online and offline versions of 15 out of the 24 meters in Table 9 in PARS[3]. For the online versions, we implement an online interface in PARS, by which an input password/dataset will be transferred to the website's server. Then, the feedback

[3]Based on our experience, several password meters cannot be implemented offline. This is because their actual strength evaluation modules are not embedded in the source code of the registration pages.

will be returned to PARS and presented to users. For the offline versions, first, we de-obfuscate and analyze the source code on the registration pages of the 15 websites. Then, we identify the specific modules for password strength evaluation within their source code (more details are in [2]).

Both online and offline implementations have their advantages. Since a site may update its password checker/meter at any time, the online implementation could be more up-to-date and thus is more accurate with respect to that site's status quo. However, large-scale online password evaluation (e.g., a large password dataset) could cause a heavy traffic load to a site's server. Therefore, in this scenario, offline evaluation is more appropriate and faster.

Commercial Meters Evaluation. Below, we evaluate the performance of commercial password meters using Yahoo!, Google, Target, and Bloomberg's meters as examples. More evaluation results of the 150 commercial password meters/policy checkers can be found at [2]. The strength evaluation results of the passwords of `CSDN`, `LinkedIn`, and `Gamigo` by the four example meters are shown in Fig.2 (a)-(d), respectively. To conduct a comparative study, we also use these four commercial meters to evaluate the cracked passwords of `CSDN` by `Duduniu`-trained OMEN, of `LinkedIn` by `Rockyou`-trained VCT, and of `Gamigo` by `Rockyou`-trained VCT. The results are shown in Fig.2 (e)-(h), respectively.

(1) From Fig.2 (a)-(d), generally, different password meters are not consistent with each other, which agrees with the observations in [24]. Google's meter labels most passwords of `CSDN`, `LinkedIn`, and `Gamigo` as strong. Particularly, under Google's meter, `CSDN` has more strong passwords than `LinkedIn` and `Gamigo`, which contradicts the cracking results in Table 5. Target's meter labels most passwords as weak or invalid, which is very different from that of the other three meters. Yahoo! and Bloomberg's meters have similar classification results, and most passwords of the three example datasets are labeled as strong/good.

(2) From Fig.2 (a) and (e), we see that most cracked passwords of `CSDN` are labeled as weak. However, there are still a considerable number of cracked passwords of `CSDN` that are labeled as strong. Furthermore, more than 62% of the cracked passwords of `LinkedIn` and `Gamigo` are labeled as strong by Yahoo!'s meter. A similar situation exists with Bloomberg's meter, which can be seen from Fig.2 (d) and (h). Therefore, to some extent, these two meters may provide incorrect feedback to users in the registration process.

(3) From Fig.2 (b) and (f), we see that about 86.8%, 33.6%, and 36.3% cracked passwords of `CSDN`, `LinkedIn`, and `Gamigo` are labeled as strong by Google's meter, respectively. Therefore, the helpfulness of Google's meter is very limited in guiding users to choose secure passwords in practice. Rather, the feedback of Google's meter may lead to vulnerable passwords.

(4) From Fig.2 (c) and (g), we see that Target's meter can properly label the cracked passwords of the three datasets. Only a very small number of cracked passwords are labeled as strong or extra strong. Therefore, compared to the other three commercial meters, Target's meter is more effective to guide users to choose secure passwords.

6. INSIGHTS AND DISCUSSION

As discussed in Section 4, no cracking algorithm is optimal in all scenarios. Therefore, to improve the performance of existing cracking algorithms, *an intuitive idea is to design an effective Hybrid Password Cracking (HPC) algorithm that combines the advantages of existing schemes*, e.g., PCFG-based schemes, Markov-based schemes. To understand the feasibility of this proposal, we conduct further analysis based on the results of different cracking algorithms in Table 5.

First, we study the following two questions: *is it reasonable and possible to design an improved HPC algorithm?* and *if it is reasonable, how much improvement can be achieved?* To answer these two questions, we define the *Relative Improvement Ratio* (RIR) of two cracking algorithms. Let X and Y be two sets of cracked passwords of two algorithms A_1 and A_2 under the same setting, e.g., the two sets of cracked `7k7k` passwords by `Renren`-trained PCFG and `Renren`-trained OMEN. The RIR of A_1 given by A_2, denoted by $\overleftarrow{A_1A_2}$, is defined as $\overleftarrow{A_1A_2} = \frac{|Y \setminus X|}{|X|}$. Therefore, RIR indicates the potential improvement of an algorithm if it incorporates the advantage of another algorithm.

In Table 10, we demonstrate the RIR values of PCFG (P), VCT (V), 3g (3), and OMEN (O) under example scenarios based on the cracking results in Table 5, e.g., .77 (in red) is the RIR of `Tianya`-trained PCFG given the advantage of `Tianya`-trained 3g when cracking `7k7k`, and 4.35 (in blue) is the RIR of `Tianya`-trained VCT given the advantage of `Tianya`-trained OMEN when cracking `7k7k`. From Table 10, we have the following observations.

(1) Every algorithm has potential improvement given the advantage of another algorithm. For instance, when cracking `LinkedIn`, the RIRs of `Rockyou`-based PCFG and `Rockyou`-based OMEN given to each other are 54% and 56%, respectively. Therefore, *the proposal of designing HPC algorithms is reasonable and promising.*

(2) The RIRs are different under different scenarios. For instance, when cracking `7k7k`, the RIRs of `Renren`-trained VCT given by `Renren`-trained 3g and `Renren`-trained OMEN are 66% and 435%, respectively, which implies VCT could benefit more from OMEN than from 3g. Therefore, it is important to consider which algorithms should be combined in designing HPC schemes.

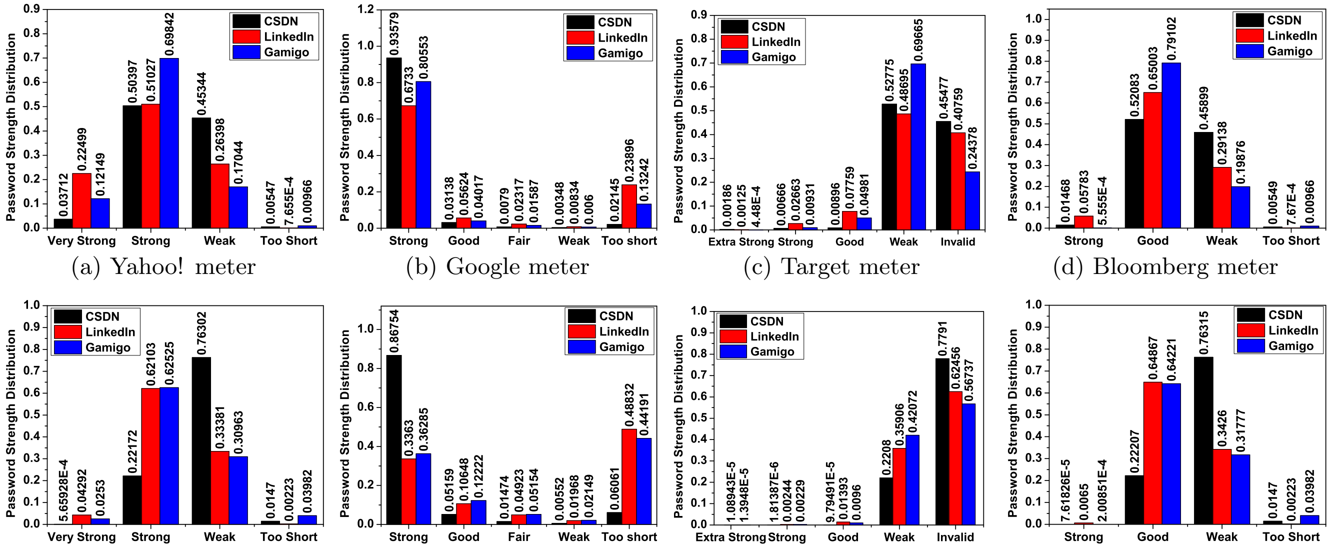
(3) The RIRs of two algorithms are asymmetric. For instance, for `Rockyou`-trained VCT and `Rockyou`-trained OMEN, when cracking `CSDN`, their RIRs are 330% and 28% respectively, while when cracking `LinkedIn`, their RIRs given to each other are 53% and 147% respectively.

According to our observations, it is reasonable and promising to improve existing password cracking algorithms by designing HPC algorithms.

Limitations. The limitations of this paper are as follows. First, due to the space limitation, we do not include all the evaluation results in the paper but we put them on the project website [2]. Second, when evaluating password cracking research, we focus on large-scale offline attacks. We do not consider other forms of attacks, e.g., phishing attacks.

7. CONCLUSION

In this paper, we propose and implement PARS, an open-source and modular password analysis and research system which provides a uniform, comprehensive and scalable research platform for password security. In addition, we propose and implement RIR, which sheds light on a future research topic of designing hybrid password cracking algorithms. Using PARS, researchers can conveniently conduct research and password security analysis, respectively, in a



(a) Yahoo! meter (b) Google meter (c) Target meter (d) Bloomberg meter

(e) Yahoo! meter (cracked) (f) Google meter (cracked) (g) Target meter (cracked) (h) Bloomberg meter (cracked)

Figure 2: Commercial meters evaluation.

Table 10: RIR analysis.

Training Data	7k7k								CSDN							
	$\overleftarrow{P3}$	$\overleftarrow{3P}$	$\overleftarrow{PO}$	$\overleftarrow{OP}$	$\overleftarrow{V3}$	$\overleftarrow{3V}$	$\overleftarrow{VO}$	$\overleftarrow{OV}$	$\overleftarrow{P3}$	$\overleftarrow{3P}$	$\overleftarrow{PO}$	$\overleftarrow{OP}$	$\overleftarrow{V3}$	$\overleftarrow{3V}$	$\overleftarrow{VO}$	$\overleftarrow{OV}$
Renren	**.77**	.45	4.63	.03	.66	.56	**4.35**	.13	.07	6.1	2	.30	.05	8.6	1.6	.56
Tianya	.79	.59	5.5	.04	.64	.76	4.75	.15	.09	6.55	2	.28	.05	11.6	1.3	.73
Rockyou	1.9	.23	19	.02	1.57	.28	16	.04	.13	5.5	2.7	.25	.15	4.7	3.3	.28
Gamigo	1.4	.59	6	.09	.69	1.03	3.17	.29	.06	15	.70	1.1	.07	12.1	.96	1

Training Data	Duduniu								LinkedIn							
	$\overleftarrow{P3}$	$\overleftarrow{3P}$	$\overleftarrow{PO}$	$\overleftarrow{OP}$	$\overleftarrow{V3}$	$\overleftarrow{3V}$	$\overleftarrow{VO}$	$\overleftarrow{OV}$	$\overleftarrow{P3}$	$\overleftarrow{3P}$	$\overleftarrow{PO}$	$\overleftarrow{OP}$	$\overleftarrow{V3}$	$\overleftarrow{3V}$	$\overleftarrow{VO}$	$\overleftarrow{OV}$
Renren	.10	3.6	.76	.48	.08	4.8	.80	.99	.17	2.6	.26	1.4	.14	3.2	.30	2.1
Tianya	.24	2.6	1.6	.34	.15	4.3	1.1	.80	.18	3.3	.25	2.4	.18	3.1	.26	2.5
Rockyou	.24	2.6	1.5	.37	.23	2.5	1.7	.51	.20	2.5	.54	.56	.13	3.9	.53	1.5
Gamigo	.10	7.3	.32	2.05	.10	7.02	.40	2.3	.06	11	.22	2.1	.04	16	.23	3.8

comprehensive and comparable way.

Acknowledgment

We are very grateful to Rafavel Veras, Markus Dürmuth, Saranga Komanduri, Xin Hu, and Tielei Wang for their help.

8. REFERENCES

[1] J. Bonneau, C. Herley, P. C. Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. *S&P*, 2012.

[2] The PARS Project. ******.

[3] Z. Li, W. Han, and W. Xu. A large-scale empirical analysis on chinese web passwords. *Usenix*, Securit.

[4] J. Bonneau. The science of guessing: Analyzing an anonymized corpus of 70 million passwords. *S&P*, 2012.

[5] S. Houshmand and S. Aggarwal. Building better passwords using probabilistic techniques. *ACSAC*, 2012.

[6] B. Ur, P. G. Kelley, S. Komanduri, J. Lee, M. Maass, M. L. Mazurek, T. Passaro, R. Shay, T. Vidas, L. Bauer, N. Christin, and L. F. Cranor. How does your password measure up? the effect of strength meters on password creation. *USENIX*, 2012.

[7] C. Castelluccia, M. Dürmuth, and D. Perito. Adaptive passwords-strength meters from markov models. *NDSS*, 2012.

[8] M. Weir, S. Aggarwal, M. Collins, and H. Stern. Testing metrics for password creation policies by attacking large sets of revealed passwords. *CCS*, 2010.

[9] P. G. Kelley, S. Komanduri, M. L. Mazurek, R. Shay, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, and J. López. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. *S&P*, 2012.

[10] J. Ma, W. Yang, M. Luo, and N. Li. A study of probilistic password models. *S&P*, 2014.

[11] W. E. Burr, D. F. Dodson, and W. T. Polk. Electronic authentication guideline. *NIST*, 2006.

[12] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang. The tangled web of password reuse. *NDSS*, 2014.

[13] M. Dürmuth, A. Chaabane, D. Perito, and C. Castelluccia. When privacy meets security: Leveraging personal information for password cracking. *CoRR abs/1304.6584*, 2013.

[14] R. Veras, C. Collins, and J. Thorpe. On the semantic patterns of passwords and their security impact. *NDSS*, 2014.

[15] M. Dell' Amico, P. Michiardi, and Y. Roudier. Password strength: An empirical analysis. *Infocom*, 2010.

[16] A. Narayanan and V. Shmatikov. Fast dictionary attacks on passwords using time-space tradeoff. *CCS*, 2005.

[17] M. Weir, S. Aggarwal, B. Medeiros, and B. Glodek. Password cracking using probabilistic context-free grammars. *S&P*, 2009.

[18] Y. Zhang, F. Monrose, and M. K. Reiter. The security of modern password expiration: An algorithmic framework and empirical analysis. *CCS*, 2010.

[19] John the Ripper-bleeding jumbo. https://github.com/magnumripper/johntheripper.

[20] M. L. Mazurek, S. Komanduri, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, P. G. Kelley, R. Shay, and B. Ur. Measuring password guessability for an entire university. *CCS*, 2013.

[21] R. Shay, S. Komanduri, P. G. Kelley, P. G. Leon, M. L. Mazurek, L. Bauer, N. Christin, and L. F. Crano. Encountering stronger password requirements: User attitudes and behaviors. *SOUPS*, 2010.

[22] S. Komanduri, R.Shay, L. F. Cranor, C. Herley, and S. Schechte. Telepathwords: Preventing weak passwords by reading users' minds. *USENIX*, 2014.

[23] A. Forget, S. Chiasson, P. C. V. Oorschot, and R. Biddle. Improving text passwords through persuasion. *SOUPS*, 2008.

[24] X. C. Carnavalet and M. Mannan. From very weak to very strong: Analyzing password-strength meters. *NDSS*, 2014.