Guessing human-chosen secrets

Joseph Bonneau

University of Cambridge
Churchill College

April 2012

This dissertation is submitted for
the degree of Doctor of Philosophy

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text.

No parts of this dissertation have been submitted for any other qualification.

This dissertation does not exceed the regulation length of $60,000$ words, including tables and footnotes.

To Fletcher.
I'm glad you're back.

—Joseph Bonneau, April 2012

Acknowledgements

I am grateful to my supervisor Ross Anderson for help every step of the way, from answering my emails when I was a foreign undergraduate to pushing me to finally finish the dissertation. He imparted countless research and life skills along the way, in addition to helping me learn to write in English all over again. I was also fortunate to be surrounded in Cambridge by a core group of "security people" under Ross' leadership willing to ask the sceptical questions needed to understand the field.

In particular, I've benefited from the mentorship of Frank Stajano and Markus Kuhn, the other leaders of the group, as well as informal mentorship from Richard Clayton, Bruce Christianson, Mike Bond, George Danezis, Claudia Diaz, Robert Watson and Steven Murdoch amongst many others. I thank Arvind Narayanan for his support and mentorship from afar. I am most appreciative of the personal mentorship extended to me by Saar Drimer through my years in the lab, which always pushed me to be more honest about my own work.

I am grateful to all of my collaborators, particularly my fellow students Andrew Lewis, Sören Preibusch, Jonathan Anderson, Rubin Xu and Ekaterina Shutova. I was also fortunate to be able to collaborate remotely with Cormac Herley and Paul van Oorschot, senior researchers who always treated me as an equal. I owe special thanks to Hyoungshick Kim, thanks to whose patience and positivity I spent thousands of hours peacefully sharing a small office.

My research on passwords would not have been possible without the gracious cooperation and support of many people at Yahoo!, in particular Richard Clayton for helping to make the collaboration happen, Henry Watts, my mentor, Elizabeth Zwicky who provided extensive help collecting and analysing data, as well as Ram Marti, Clarence Chung, and Christopher Harris who helped set up data collection experiments. My research on PINs depended on many people's help, including Alastair Beresford for assistance with survey design, Daniel Amitay for sharing data, and Bernardo Bátiz-Lazo for comments about ATM history.

I never would have made it to Cambridge without many excellent teachers along the way. From Stanford, I thank Ilya Mironov, Dan Boneh, and John Mitchell for inspiring me to pursue computer security research as an undergraduate. I thank Robert Plummer for his

mentorship, for inspiring me to love teaching and encouraging me to study at Cambridge. From earlier on, I thank all of the teachers who showed me how to learn: Mike Kelemen, Steve Hettleman, David Goldsmith, David Goldman, Michael Collins, and David Nelson.

My research depended on a large suite of free software. I am indebted to the entire free software movement, in particular the developers of the GNU, Linux, Ubuntu, GNOME, Mozilla, TEX/LATEX, Python, matplotlib, SciPy, NumPy, and R projects.

My time in Cambridge was supported financially by the Gates Cambridge Trust. I am particularly grateful to Gordon Johnson and James Smith for personal help and encouragement, as well as all the officers of the Gates Scholars' Council during my time as president.

I thank all of my friends in Cambridge for helping me adjust to life in a new country and the frustrations of life as a graduate student. I'll particularly remember my housemates Andrew Marin, Niraj Lal and Matt Warner, as well as Andra Adams, Marianne Bauer, Lindsay Chura, Justine Drennan, Molly Fox, Talia Gershon, Simone Haysom, Julia Fan Li, Stella Nordhagen, Adeline Oka, Sri Raj, Megan Sim, Jessica Shang, Brian Spatocco, Elsa Treviño, and Cleo Tung for close friendships, all of which turned a day around for me at some point during my time in Cambridge.

Thanks to modern technology I also received considerable support from friends overseas which kept my spirits up throughout my time in England. I thank my friends Alexandra Bowe, Dave Emme, Alissa Chow and Brent Newhouse for being there when I wanted to talk to a familiar voice, as well as the entire Smitty league of Keegan Dresow, Will Helvestine, Tyler Jank, Jon Levine, Bobby Simon and Steve Zabielskis for listening to my rants and giving me a reason to laugh just about every day.

Above all I am grateful for support from my family, who may be few in number and small in stature but have remained a big presence in my life through it all: my cousins Selim and Sinan, uncle Turhan and aunt Phyllis for welcoming me in Turkey after my years-delayed trip, my aunt Amy for chocolate and weekly trivia questions, my grandmother Anne for making sure I keep warm, my grandmother Margaret for teaching me to love words, my siblings Buzzy and Alissa for making sure I can laugh at myself, and my mother and father for giving me so much and teaching me to always appreciate it. I love you all.

Guessing human-chosen secrets

Joseph Bonneau

Summary

Authenticating humans to computers remains a notable weak point in computer security despite decades of effort. Although the security research community has explored dozens of proposals for replacing or strengthening passwords, they appear likely to remain entrenched as the standard mechanism of human-computer authentication on the Internet for years to come. Even in the optimistic scenario of eliminating passwords from most of today's authentication protocols using trusted hardware devices or trusted servers to perform federated authentication, passwords will persist as a means of "last-mile" authentication between humans and these trusted single sign-on deputies.

This dissertation studies the difficulty of guessing human-chosen secrets, introducing a sound mathematical framework modeling human choice as a skewed probability distribution. We introduce a new metric, α-guesswork, which can accurately models the resistance of a distribution against all possible guessing attacks. We also study the statistical challenges of estimating this metric using empirical data sets which can be modeled as a large random sample from the underlying probability distribution.

This framework is then used to evaluate several representative data sets from the most important categories of human-chosen secrets to provide reliable estimates of security against guessing attacks. This includes collecting the largest-ever corpus of user-chosen passwords, with nearly 70 million, the largest list of human names ever assembled for research, the largest data sets of real answers to personal knowledge questions and the first data published about human choice of banking PINs. This data provides reliable numbers for designing security systems and highlights universal limitations of human-chosen secrets.

Contents

Computers are useless. They can only give you answers.

—Pablo Picasso, 1968

Chapter 1

Introduction

Secret knowledge stored in human memory remains the most widely deployed means of human-computer authentication. Most notably, text passwords dominate authentication over the Internet and numeric PINs dominate authentication for payment card transactions. Most security engineers believe both are weak points whose security continues to decline with the increasing number of third parties seeking to authenticate users [121, 244].

Reliable data on damages caused by weak human-chosen secrets is hard to come by [141], but a recent study of corporate data breaches commissioned by Verizon [12] suggested that nearly a third are due to stolen login credentials. This surpasses classic technical exploits such as SQL injection or buffer overflows. Of attacks using stolen login credentials, over a quarter were estimated to be stolen by some form of a guessing attack. Guessing attacks have had major business implications, such as a 2009 incident in which a vandal guessed a Twitter executive's password and was able to leak all of the company's internal documents [75].

There also exists a significant threat to individuals' private online accounts. A 2008 study by Symantec [7] of online black markets found a vibrant economy trading in stolen passwords. Due to the widespread re-use of passwords across sites [152], an emerging attack model is to compromise accounts by a guessing attack against a low-security website and attempt to re-use the credentials at critical websites [240].

At the same time as attacks on passwords are becoming an industrial-scale threat, the past few years have seen massive data sets of passwords available for study for the first time. While passwords and PINs have long been considered weak secrets, computer security researchers have lacked a standard way for analysing how resistant these credentials actually are to guessing. The literature lacks sound methodology to answer simple questions such as "Do passwords provide better security against guessing than PINs?" or "Do users of website A pick more secure passwords than users of website B?".

This dissertation aims both to rigorously define an appropriate framework for answering these questions and to introduce the largest data sets of human-chosen secrets yet published to provide standard benchmarks for security engineers to use when designing systems which incorporate human choice.

1.1 Model of authentication and guessing attacks

The first step in any security analysis is defining what we are trying to protect and what attackers are capable of [22]. We introduce an abstract model of authentication with just enough formalism to capture all of the important practical cases of human-computer authentication. Many of the concepts we introduce will be described in detail later, which we indicate by including the section number, i.e. (§1.1).

1.1.1 Basic authentication protocol

In cryptographic terms, authentication is a protocol between a *principal* claiming a certain identity, called the *prover* or *claimant*, and a sceptical principal called the *verifier* requesting proof. In remote human-computer authentication, the prover is often called the *user* and the verifier called the *server*. Software operating on behalf of the user is called the *user agent* or *browser*. The term *client* may refer to either the user, the user agent, or their combination.

In *static* authentication protocols, the prover sends a *password* $\mathbf{x}$,[1] also called a *secret* or *token*, to the verifier along with a claimed identity[2] $\mathbf{i}$. We will often use the term password in a generic sense to refer to any fixed secret knowledge, which may be a PIN (§4), graphical password (§2.3.2), or some other item. When clarity is needed, we will use the term *text password* to mean a traditional short, human-chosen character string.

Upon receipt of an authentication request $(\mathbf{i}, \mathbf{x})$ the verifier checks it against a database, ideally in hashed format (§2.2.1), and makes an authentication decision. Because the prover must be granted or denied access the verifier acts as an *oracle* to test whether arbitrary pairs $(\mathbf{i}, \mathbf{x})$ are valid. Together, a valid pair $(\mathbf{i}, \mathbf{x})$ is referred to as a *credential*. An attacker's goal is typically to obtain one or more valid credentials $(\mathbf{i}, \mathbf{x})$ from a target verifier.[3]

[1] We use $\mathbf{x}$ to denote a password instead of p to avoid confusion with probabilities.

[2] The term *identity* is complicated and no universal definition exists. For our purposes, we'll broadly say that an identity is any string such as a name, email address or username which links actions taken at two different points in time to the same principal.

[3] Note that verifiers need not grant access in all cases given a valid credential. They may limit a credential to certain machines or IP addresses, for example, as part of an *intrusion detection system* designed to mitigate credential theft [82]. We consider this out of scope.

Challenge-response authentication

Some knowledge-based authentication schemes require the verifier to send a specific *challenge* $\mathbf{c}$ to the prover prior to receiving the password. In some schemes $\mathbf{c}$ is fixed for each prover, such as a personal knowledge question (§7) or an image on which to click secret points (§2.3.2). A fixed challenge is usually called a *prompt*.

In other schemes, referred to as *challenge-response protocols*, $\mathbf{c}$ is unique for each authentication attempt and is called a *nonce*, *timestamp* or *counter*. Varying challenges must be incorporated by responding with some function $f(\mathbf{x}, \mathbf{c})$ instead of simply $\mathbf{x}$. The computed value $f(\mathbf{x}, \mathbf{c})$ is often called a *one-time password*. The function f might be executed by a computer, in which case it can provide cryptographic security (§2.3.1), or designed to be simple enough for humans to compute with sub-cryptographic security (§2.3.2).

1.1.2 Guessing attacks

Any attacker attempting to find credentials by guessing likely passwords can be considered a *guessing attacker*. An *online guessing attack* consists of submitting guessed credentials $(\mathbf{i}, \mathbf{x})$ to the verifier to test their validity. A well-designed verifier will employ *rate-limiting* techniques (§2.2.3) to limit the number of guesses which can be made, for example by limiting the number of authentication attempts in a given time period, forcing a user to reset his or her password after too many failed attempts, or requiring the prover to solve puzzles such as CAPTCHAs.

In an *offline guessing attack*, the attacker has obtained some value cryptographically derived from $\mathbf{x}$ which can be used to verify guesses. Often this is the *password hash* $\mathbf{H}(\mathbf{x})$ obtained through a database compromise, but it may also be the value $f(\mathbf{x}, \mathbf{c})$ for a known challenge c obtained by eavesdropping. No rate-limiting is possible in this scenario so the attacker is only throttled by available computational resources. Offline attacks are also called *brute-force attacks* or *password cracking* (§2.4).

We may also classify guessing attacks by the attacker's goals. In a *targeted* or *vertical* attack, the attacker only seeks to determine $\mathbf{x}$ for a fixed value of $\mathbf{i}$. Targeted attackers can research the targeted user $\mathbf{i}$ to enhance their guessing strategy or attempt to steal $\mathbf{x}$ outside of the authentication protocol completely.

In a *trawling* or *horizontal* attack, an attacker has a large list of identities $\mathbf{i}_1, \ldots, \mathbf{i}_k$ and is interested in discovering the correct $\mathbf{x}$ for as many as possible. A trawling attacker typically won't have user-specific information for any of the available identities and will instead guess the most likely population-wide passwords. Security economics suggests that on the Internet, trawling attacks scale more efficiently than targeted attacks [139].

Preventing the attacker from assembling a large list of valid identities is one defence against a trawling attack. An adversary may attempt to test if an identity $\mathbf{i}$ is valid without knowing

the correct password **x** to prevent wasted guessing effort. This is referred to as *user probing*. A securely implemented verifier should return a generic error if either the identity **i** does not exist or the wrong **x** is supplied to prevent user probing.[4]

While all combinations of offline/online and targeted/trawling attack are possible, some literature assumes that online attacks are always trawling attacks and offline attacks are always targeted which are the most common cases.

1.1.3 Other attacks

An attacker may steal credentials without guessing them. One approach is to masquerade as a valid verifier and attempt to collect credentials when provers attempt to authenticate. This is called *phishing*.[5] Alternately, an attacker may try to observe authentication between a valid prover and verifier. The most common mechanism is by running malicious software (or *malware*) on a victim's computer which silently records credentials as they are entered, often called a *keylogging attack*. Credentials can also be observed during transmission if they are not encrypted in an *eavesdropping attack* (§2.2.2), also referred to as *password sniffing* or *password snooping*. Finally, an attacker may physically observe a user entering credentials in a *physical observation attack*. This is often referred to as *shoulder-surfing* if the attacker personally observes the entry of credentials, but a physical observation attack may also involve video cameras or other automated equipment.

This dissertation will focus exclusively on guessing and not consider these attacks any further.

1.2 Outline of this dissertation

We'll begin in §2 with an overview of the history of passwords, PINs and other secret-knowledge based authentication systems. This chapter will focus particularly on why efforts to replace these schemes have historically failed and why past analysis of guessing difficulty has been ad hoc and unsatisfactory.

In §3 we'll explore mathematical metrics of guessing difficulty, quickly surveying previously proposed measures and introducing new partial guessing metrics which accurately model an attacker willing to give up on difficult values. The chapter builds towards a new metric, α-guesswork, which is a generalisation of the traditional guesswork metric which can be parameterised to capture realistic attack scenarios. Several properties of the new metric are proved which demonstrate why old metrics can be arbitrarily inaccurate. The concept of a guessing curve is introduced to visualise a continuum of guessing attacks and their difficulty.

[4] In [47] we found that 96% of websites enable user probing by simple means.

[5] Dhamija et al. provide a good overview of phishing [88].

The new partial guessing metrics are applied in §4 to analyse numeric PINs, a relatively easy application since banking PINs are chosen from a small finite domain of possibilities. This section utilises some leaked data sets and a large user survey to introduce the first-ever estimates of how difficult real banking PINs are to guess.

In §5 we'll tackle the difficulty of estimating guessing metrics without perfect knowledge of the distribution but instead with a large set of random samples. Even at massive sample sizes it can be impossible to accurately compute many guessing metrics for some distributions of interest, particularly passwords. We'll then introduce methodology for estimating which guessing metrics can be approximated without assuming anything about the underlying distribution of secrets, as well as a parametric method for fitting observations to a model distribution which shows promise in the case of password distributions.

This allows studying the guessing difficulty of passwords empirically in §6. First, a novel experimental setup is described for collecting a large password data set in a privacy-preserving manner, enabling the calculation of mathematical guessing metrics which only depend on the probability distribution of passwords and not their semantic content. This approach was applied in a large experiment at Yahoo!, collecting data representing almost 70 million users' passwords. Demographic subpopulations within the Yahoo! data can be compared to one another and the entire data set is compared to other sources of password data and distributions of words in English text.

In §7, the guessing difficulty of personal knowledge questions is studied, again using a large data set collected at Yahoo! of dozens of questions in different languages. This data is also compared to published statistics about the distribution of human names, as well as a large data set crawled from the online social network Facebook, to evaluate how real users often do not answer questions as they were intended.

Finally, two related guessing problems are studied. §8 introduces metrics for the loss of efficiency when an attacker chooses values to guess based on a different distribution instead of perfect knowledge of the actual distribution under attack. §9 introduces some simple metrics for analysing the guessing difficulty of a single item drawn from a distribution, instead of analysing the complete distribution. Some concluding perspectives are provided in §10 which discuss implications for future research on human authentication.

1.3 Prerequisites

This dissertation is written for readers with a basic background in probability theory for understanding concepts such as discrete and continuous probability distributions, the expectation, variance, and standard error of a random variable, and elementary probability distributions such as the uniform, binomial and Poisson. We recommend the following textbook, now freely available online, for an introduction to these topics:

- Charles M. Grinstead and J. Laurie Snell. *Introduction to Probability.* American Mathematical Society, Providence, 2nd edition, 1997

This dissertation requires no specific background in information theory, computer security or cryptography and will aim to introduce and define all concepts from these fields as they arise. Still, we can recommend the following two textbooks, both also freely available online, for readers with no background in these subjects:

- David J.C. Mackay. *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press, Cambridge, 2003
- Ross J. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems.* Wiley, New York, 2nd edition, 2008

1.4 Mathematical notation

This dissertation will aim to follow standard notation where possible. Because the literature on probability varies significantly we summarise our notation here, chosen to allow concise reasoning about properties of *discrete probability distributions* (or *categorical distributions*) typically subject to guessing. A glossary of all symbols used in the text is provided in §A.

Probability distributions

We denote a distribution with a calligraphic letter, such as $\mathcal{X}$. We use lower-case x to refer to a specific event in the distribution, for example an individual password. The probability of x is denoted as p_x. Formally, $\mathcal{X}$ is a set of events $x \in \mathcal{X}$, each with an associated probability $0 < p_x \leq 1$, such that $\sum p_x = 1$. We assume all events are disjoint and have non-zero probability. We use N to denote the total number of events in $\mathcal{X}$ with non-zero probability, which we can also denote $|\mathcal{X}|$.

We often refer to events by their *index*, usually written i, with the most-probable event having index 1 and the least probable having index N. We refer to the i^{th} most common event as x_i and call its probability p_i. Thus, the probabilities of the events in $\mathcal{X}$ form a monotonically decreasing sequence $p_1 \geq p_2 \geq \ldots \geq p_N$.

We denote an unknown variable as X, denoting $X \xleftarrow{\text{R}} \mathcal{X}$ if it is drawn at random from $\mathcal{X}$.

A *discrete uniform distribution* $\mathcal{U}_N$ is one in which all N events are equally probable, that is $\forall_{1 \leq i \leq N} \quad p_i = \frac{1}{N}$.

A *mixture distribution* $\mathcal{Z}$ is a distribution composed of two or more constituent probability distributions $\mathcal{Z}_i$, each with an associated probability q_i. To draw an event $Z \xleftarrow{\text{R}} \mathcal{Z}$, one first randomly chooses one of the constituent distributions $\mathcal{Z}_i$ according to q_i and then chooses $Z \xleftarrow{\text{R}} \mathcal{Z}_i$. We can denote a mixture distribution's decomposition as $\mathcal{Z} = q_1 \cdot \mathcal{Z}_1 + q_2 \cdot \mathcal{Z}_2 + \ldots$.

Logarithms

We frequently use logarithms when defining guessing metrics. In all cases we use $\lg x$ to denote the base-2 logarithm of x, which is the preferred base for security engineering. In §3.2.4 we discuss the possibility of using a different base when computing guessing metrics.

Passwords and data sets

An example text password is denoted as `password`. Similarly, we will refer to a specific 4-digit PIN as `2012`. When we refer to a specific real-world data set, such as the list of passwords leaked from the website RockYou, we denote the data set as RockYou. A list of password data sets, with details about their provenance, is provided in §D.

1.5 Previous publications and collaboration

This dissertation incorporates text and concepts directly from several of my previous publications. Significant parts of the guessing model introduced in §3, statistical model in §5 and password data presented in §6 and §8 were packaged as a conference publication, written by myself alone with assistance from Yahoo! staff for data collection:

- Joseph Bonneau. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *SP '12: Proceedings of the 2012 IEEE Symposium on Security and Privacy*, 2012

The data on PINs in §4 was published in collaboration with Sören Preibusch, who took the lead on survey design and deployment, and Ross Anderson who assisted with survey design and historical background (§2.1.2):

- Joseph Bonneau, Sören Preibusch, and Ross Anderson. A birthday present every eleven wallets? The security of customer-chosen banking PINs. In *FC '12: The 16th International Conference on Financial Cryptography and Data Security*. Springer-Verlag, 2012

Some data on personal knowledge questions analysed in §7 was published in collaboration with Mike Just and Greg Matthews, who assisted in gathering government census records from around the world:

- Joseph Bonneau, Mike Just, and Greg Matthews. What's in a name? Evaluating statistical attacks against personal knowledge questions. In *FC '10: The 14th International Conference on Financial Cryptography and Data Security*. Springer-Verlag, 2010

The model for evaluating the strength of individual elements from a distribution presented in §9 was previously published, written by myself alone:

- Joseph Bonneau. Statistical metrics for individual password strength. In *20th International Workshop on Security Protocols*, 2012

Some data presented in §6.4 on distributions of words and phrases in English was published in collaboration with Ekaterina Shutova, who introduced me to the linguistic data sets and suggested their application to modeling passphrases:

- Joseph Bonneau and Ekaterina Shutova. Linguistic properties of multi-word passphrases. In *USEC '12: Workshop on Usable Security*, 2012

Finally, the background material in §2 adapts material and concepts from all of the above publications, in addition to the following other publications on password security written during the course of my doctoral research. All of the text here is my own, though I am indebted to Ross Anderson, Cormac Herley, Paul van Oorschot, Sören Preibusch and Frank Stajano for general collaboration and conversation about the password security field.

- Joseph Bonneau, Cormac Herley, Paul C. van Oorschot, and Frank Stajano. The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In *SP '12: Proceedings of the 2012 IEEE Symposium on Security and Privacy*, 2012
- Joseph Bonneau. Getting web authentication right: a best-case protocol for the remaining life of passwords. In *19th International Workshop on Security Protocols*, 2011
- Joseph Bonneau and Sören Preibusch. The password thicket: technical and market failures in human authentication on the web. In *WEIS '10: Proceedings of the 9th Workshop on the Economics of Information Security*, 2010

1.6 Statement on research ethics

This dissertation uses data about secrets chosen by millions of real people using deployed systems. Using such data carries valid ethical concerns [96] and care was taken to ensure that no activities undertaken for research made any user data public which wasn't previously. This data was obtained, as noted throughout the text, by a mixture of privacy-preserving experiments, surveys with informed consent, and analysis of data leaked publicly by security breaches. Oversight was provided by the Ethics Committee of the University of Cambridge Computer Laboratory as well as the Yahoo! legal team for data collected there.

It takes continued ingenuity to keep up with prevailing silly practices in choosing passwords.

—Fred Grampp and Robert H. Morris, 1984 [128]

Chapter 2

Background

Authentication has been studied by cryptographers, security engineers, human-computer interface designers, linguists, ethnographers, and others. This chapter will survey the diverse academic literature with particular focus on the security research motivating this dissertation.

2.1 History

The use of secret words to authenticate humans has ancient origins. The concept dates at least as far back as the military of ancient Rome, which developed a careful procedure for circulating daily *signa* or "watchwords" to prevent infiltration as documented by the historian Polybius in 118 BCE [237]. It also appears in folklore, famously in the tale of Ali Baba and the forty thieves (first translated into English in 1785 [296]), with the protagonist using the phrase "open sesame" to unseal a magical cave. Ominously, Ali Baba's greedy older brother Qasim forgets this password during the course of the story with disastrous consequences.

2.1.1 History of computer passwords

With the development of the first multi-user computer operating systems in the early 1960s, human-computer authentication was needed for the first time to prevent unauthorised access to other users' files. The Compatible Time-Sharing System at MIT [266] is often considered the first computer system to deploy passwords, storing a password for each account in an unencrypted master file. The primary security threat was users stealing scarce computing time rather than secret data [270]. Indeed, Alan Scherr admitted to committing the likely first-ever password compromise as a doctoral student in 1962 to increase the computing time available for his own jobs [303]. The CTSS implementation also saw the first-ever password database

leak in 1965 when a bug sent the password file to a public printer, requiring administrators to reset every users' password by hand [303].

Multics brought the first commercial deployment of a secure time-sharing operating system in 1968, though by 1974 designers concluded that passwords were "surprisingly easy to guess" [257]. Based on the Multics experience, Morris and Thompson improved password hashing and introduced per-user salts during the development of UNIX. They also cracked over 80% of accounts in the first published dictionary attack in 1979, warning that user-chosen passwords were a major vulnerability. A password-cracking club, the 'Computer Freaks,' arose as early as 1981 [70]. Password insecurity first gained widespread notoriety in 1988 with the launch of the infamous Morris worm,[1] which guessed passwords on every reachable host using a 431-word dictionary.

The publicity surrounding the worm and the onset of the World Wide Web in the early 1990s motivated a surge of research into replacing or improving passwords for remote human-computer authentication. Proposals for cryptographic password verification protocols [31, 32, 318] and proactive password-checking systems [36, 77, 33] saw little adoption though and text passwords quickly proliferated as the dominant means of authentication on the web. Neither Microsoft's propriety Passport system [177] nor the community-driven open-source OpenID project [246] succeeded in bringing federated authentication to the web, while graphical and cognitive authentication schemes have failed to gain significant adoption. Passwords have garnered publicity again in the past few years with the first massive leaks of password databases.

2.1.2 History of banking PINs

Contemporaneous with the development of passwords, numeric PINs (personal identification numbers) first appeared in the 1960s in automated dispensing systems at petrol filling stations [29]. Two competing British cash machines deployed in 1967 represent the first use of PINs for banking, with 6-digit PINs in the Barclays-De La Rue system rolled out in June and 4-digit PINs in the National-Chubb system in September. According to John Shepherd-Barron, leader of the De La Rue engineering team, after his wife was unable to remember six random digits he reduced the length to four. Interestingly, cash machines deployed in Japan and Sweden in 1967 used no PINs and absorbed losses from lost or stolen cards. As late as 1977, Spain's La Caixa issued cards without PINs.

PINs were initially bank assigned by necessity as they were hard-coded onto cards using steganographic schemes such as dots of carbon-14. Soon a variety of schemes for storing a cryptographic transformation of the PIN developed.[2] The IBM 3624 ATM controller [41]

[1] The Morris worm was written by Robert T. Morris, the son of Robert H. Morris who first studied dictionary attacks.

[2] James Goodfellow patented a cryptographic PIN derivation scheme in 1966 [151]. Amongst others, he has been called the inventor of PINs and ATMs.

introduced an influential scheme in 1977 which first allowed easy changes to customers' PINs. Banks gradually began allowing customer-chosen PINs in the 1980s as a marketing tactic. Most modern cards use the Visa PVV scheme, which stores a DES-based encryption of the account number and PIN [41].

The interconnection of ATM networks globally in the 1990s cemented the use of PINs for payment card authentication with the 1993 ISO 9564 standard [9] and 1995 EMV standard [6]. The EMV protocol has given PINs the further role of authorising payments at merchant tills, with chipped cards verifying the customer's PIN internally,[3] requiring PINs to be entered more often and in a plethora of terminals. This use of PINs remains technically distinct from ATM authentication, though all practical deployments have used a single PIN for both purposes.

Chipped cards have also enabled the deployment of hand-held Chip Authentication Program (CAP) readers since 2008 for verifying Internet transactions [93]. CAP readers allow muggers to verify a PIN demanded from a victim during an attack; they can also be used to guess offline the PIN on a found or stolen card.

Curiously, while 4-digit human-chosen PINs predominate in English-speaking countries, in other locales card issuers still require longer PINs. For example, banks in Switzerland assign 6–8 digit PINs and banks in Italy typically use 5-digit PINs. Canadian banks use a mixture of 4-digit and 6-digit PINs. Most banks now allow user-chosen PINs, with a few regional exceptions such as Germany.[4]

2.2 Practical aspects of password authentication

2.2.1 Password hashing

Roger Needham and Michael Guy are credited with first proposing the one-way scrambling of stored passwords in the 1963 Titan system [314]. By storing the result of a one-way function $\mathbf{H}(\mathbf{x})$, instead of simply $\mathbf{x}$, and recomputing $\mathbf{H}(\mathbf{x}')$ for any submitted password $\mathbf{x}'$, theft of the password file does not reveal plaintext passwords. In the absence of standard cryptographic hash functions, several ad hoc algorithms were proposed [161, 242]. The proprietary scheme deployed by Multics was broken by Downey in 1974 [91]. This motivated Morris and Thompson's development of the UNIX `crypt()` function [212], consisting of 25 iterations of a tweaked DES cipher[5] and 12-bit random values for each user called *salts*, with passwords limited to 8 ASCII characters. This has influenced most designs since[6] and is still occasionally used on the web; it was ultimately proved cryptographically secure in 2000 [302].

[3] EMV was deployed in the UK from 2003 under the branding "Chip and PIN." It is now deployed in most of Europe but notably not in the United States.

[4] As of 2012, several German banks have begun to allow human-chosen PINs.

[5] DES, the Data Encryption Standard, was first published in 1977 [2].

[6] A notable exception is Microsoft's LM hash algorithm, deployed through the 1990s and still occasionally used for compatibility, which uses no salting or iteration and hashes two halves of the password separately [72].

Feldmeier and Karn noted in 1989 [99] that increasingly fast implementations of `crypt()` would soon allow brute-forcing the entire input space. Manber proposed increasing the cost of brute-force by using secret salt values in 1996 [197] which must be brute-forced during verification. Kelsey et al. formally studied the *key strengthening* problem in 1997 and advocated increased iterations and mixing the salt into every round [170]. Provos and Mazières implemented Kelsey et al.'s approach with `bcrypt()` in 1999 using a parameterised iteration count [241]; this approach has been widely adopted and was standardised as the Password-Based Key Derivation Function (PBKDF2) in 2000 [164].

Despite this broad literature, Falk et al.'s 2008 study observed over 31% of banking websites failing to hash passwords [98]. We estimated a rate of 29–50% in our survey [47].

2.2.2 Password entry and session management

Authentication on the web requires implementing a session management layer on top of HTTP.[7] Fu et al. surveyed the format of cookies used by common websites in 2001 [113] and found a number of cryptographic flaws which allowed forging session cookies. Using TLS[8] to encrypt passwords during transmission is also critical, though Falk et al. found 30% of financial websites failing to do so properly [98]; we found an even higher rate of 41% for general websites [47]. Both practices enable attackers to bypass password authentication by *session highjacking* (or *sidejacking*), a risk which gained prominence in 2011 through the Firesheep tool [59]. Murdoch defined a secure cookie format resistant even to leaks of the server's database in 2008 [215]. Adida proposed a protocol to prevent cookie theft without TLS using JavaScript and HTML5 in 2008 [17]. We defined an improved cookie protocol incorporating both proposals and other practical tweaks in 2011 [45].

2.2.3 Rate limiting

Servers can mitigate online guessing attacks by limiting the rate at which authentication can be attempted. Practical difficulties include ensuring valid users have enough chances to authenticate if they forget their password or mistype it [54] and preventing denial-of-service attacks against legitimate users. Dehnad suggested making login success probabilistic in order to slow down guessing attacks [80], but frustration for legitimate users likely prohibits this approach. Pinkas and Sander instead suggest deploying CAPTCHAs to slow guessing attacks [235]. Alsaleh et al. refined this into a probabilistic protocol [19]. An alternate approach is to deploy a large number of honeypot accounts to trace guessing attackers [140].

[7] Most web browsers automatically support password sessions via the HTTP Basic and Digest Authentication mechanisms [111] though these are rarely used in practice.

[8] Transtport Layer Security (TLS) is the standard link-level encryption protocol for the Internet [89]. It still frequently called SSL after its very similar predecessor, the Secure Sockets Layer protocol.

2.2.4 Password requirements, proactive checking and blacklisting

Assisting users in creating strong and memorable passwords can be a useful part of any password implementation. The 1985 FIPS standard on password usage [3], developed from earlier military security documents, enumerated many aspects of password security policy including requirements for password length and composition, regular password updates and prohibitions against writing down passwords, sharing them, or entering them at untrusted terminals. These recommendations appear to have been adapted directly to civilian, corporate environments with little evaluation of human factors; for example Jobusch and Oldehoeft's large 1989 survey recommended requiring password changes and complex password requirements if user-chosen passwords were allowed at all [159, 160].

An influential series of studies by Adams and Sasse in the 1990s [15, 14, 259] found that many of these policies were causing significant frustration for users and were often circumvented in practice when they got in the way of users' primary tasks. Forced password changes in particular were cited as a needless cause of frustration to which many users responded by choosing trivial modifications of old passwords [14]. Zhang et al. demonstrated the ease of cracking passwords given a user's previous choices in 2010 [326]. Password ageing is now very rare in web authentication [47].

Only recently have large-scale studies [273, 169, 168] been conducted examining the effect of password composition policies, such as a minimum length or requiring certain types of characters, suggesting that length requirements are the most effective and least burdensome policy. In practice, there is little consensus among websites deploying passwords. Furnell examined the password policies of ten major websites in 2007 and found every one was unique, with only a 6-character minimum being common to the majority of sites [115]. Florêncio and Herley completed a larger study in 2010 [105], examining the password requirements of 75 commercial, government, and education websites. They found similar wide variation, which was not correlated with security requirements, and concluded that strict password policies resulted at sites with no competition such as government or university services.

Orthogonal to explicit composition policies, weak passwords can be prevented at registration by a proactive password-checking system. Several approaches were implemented in the late 1980s for UNIX. Bishop provides an early survey [37]. A large number of more complicated schemes have been proposed since [36, 77, 33, 321, 68], modeling the structure of user passwords to detect weak ones automatically (§2.5.2). In practice few websites block passwords based on proactive checks, though some websites do use *persuasion* policies that do not restrict users' choices but encourage stronger passwords through graphical or other feedback [311, 71]. The effectiveness of these methods remains unproven.

Finally, blacklisting known-weak passwords is an idea dating at least to Spafford's 1992 OPUS prototype [279], which stored a Bloom filter of passwords already registered in the system in order to ban popular choices. This concept was developed by Schechter and Herley in 2010

using a counting Bloom filter to limit the maximum frequency of any single password without outright bans [263]. This method is more flexible than model-based proactive checkers but struggles to prevent weak passwords until a large body of data exists.

2.2.5 Backup authentication for password reset

Because passwords are frequently forgotten, any practical deployment on the web must provide a mechanism to reset passwords when needed. Florêncio and Herley's study estimated that over 1% of users of a typical website forget their password and reset it each month [103].

Backup authentication is a challenging security problem requiring high reliability with failures independent of forgotten passwords. It must also be easy to use despite being rarely practised. Ideally it will also be more secure than passwords as any security weaknesses can be used to undermine security of the entire authentication system. Relative to passwords, the only sacrificed requirement is efficiency since backup authentication is used less often.

Reeder and Schechter provide a survey of backup authentication [247]. We found in our survey [47] that nearly all websites use one of two methods that we focus on: email as a secondary channel (92% of sites) and personal knowledge questions (17% of sites, including some requiring both). Jin et al. estimated these frequencies at 98% and 12%, respectively [158]. Since all known schemes have security weaknesses, it has been suggested to offer multiple schemes with user control over which combination of schemes is sufficient to re-gain access [264].

Email-based authentication

The dominant method of password reset is sending one-time passwords to a user's registered email (SMTP) address. Garfinkel argued that the practice is a weak form of PKI [119], with email addresses serving as public identifiers and email providers acting like certificate authorities. Email-based authentication can also be considered a form of single sign-on (§2.3.4), as it effectively delegates authentication to the user's email provider.

This practice arose organically with the web and was already widespread at the time of Garfinkel's 2003 survey. It has received little security research since. Van der Horst and Seamons proposed a standardised protocol for achieving this which would enable browser automation of the process [300]. Karlof et al. examined email-based authentication in a user study but in the context of user registration instead of password reset. They were able to steal secret codes sent out by email from over 40% of users via social engineering [165]. Jin et al. conducted an empirical study specifically of email-based reset and highlighted several practical weaknesses, such as users backing up passwords using email accounts relying on the same passwords [158] or relying on expired email accounts.

Personal knowledge questions

Questions based on personal knowledge (classically, "What is your mother's maiden name?") are considerably more memorable than passwords remembered specifically for authentication [328, 57, 238]. They remain widely used in banking [162] as well as for web authentication; they have also been proposed for cryptographically backing up passwords by deriving strong keys from a large number of questions [97, 112].

Security research has highlighted many weaknesses. An empirical study by Rabkin of questions used in practice found that 40% had trivially small answer spaces and 16% had answers routinely listed publicly in online social-networking profiles [243]. Even if users keep data private on social networks, inference attacks enable approximating sensitive information from a user's friends [193]. Other questions can be found in publicly available records. For example, at least 30% of Texas residents' mothers' maiden names can be deduced from birth and marriage records [129]. Social engineering attacks can be very successful in extracting users' answers. Karlof et al. were able to extract answers to personal knowledge questions from 92% of users via email phishing in a 2009 study [165]. User-chosen questions appear to be less secure than system-chosen questions; Just and Aspinall found that the majority of users choose questions with trivially few plausible answers [163].

Guessing attacks, particularly by acquaintances and family members, are also problematic. Schechter et al. found in a laboratory study that acquaintances could guess 17% of answers correctly in five tries or fewer [261], confirming similar results from earlier user studies [132, 238]. Schechter et al. also achieved success against 10% of accounts with a simple statistical guessing attack. We'll analyse guessing attacks against personal knowledge questions in §7.

Jakobsson et al. proposed querying a large number of binary preferences for items like "rap music" with some error tolerance to improve security [155, 156]. Nosseir et al. suggested querying users' browsing history or location history to generate harder-to-guess questions [223], though such an approach appears to inevitably leak private data. A related proposal specifically for social networks is to require users to identify friends in tagged photographs [324], though the preponderance of publicly available data from social networks and face recognition software seriously harm the security of this scheme [171].

Social re-authentication

An alternative academic proposal which has yet to gain large-scale deployment is requiring users to select a set of trusted friends to ask as delegates. In the event of a forgotten password, the user must contact a designated threshold of these delegates to receive one-time tokens from the server. Brainard et al. first proposed this idea in 2006 as "vouching-based" authentication [51]. A follow-up usability study by Schechter et al. found that only about 71% of participants could execute this scheme correctly and social-engineering attacks worked

against about 10% of users [262]. It has been suggested that in place of an explicit user-conducted protocol, authentication could be performed automatically by communicating with the mobile devices of nearby users to establish a user's social context [110].

2.2.6 User password management and behaviour

The difficulty for users of managing passwords was first studied at length by Adams and Sasse in 1999 [14], who observed many employees writing passwords on post-it notes on their workstations and yielding their passwords to simple social engineering attacks. They concluded that users were overwhelmed by security requirements and were placing a significant strain on IT help desks [259]. A number of follow-up studies have found users similarly unable to manage passwords for their personal computing needs [90, 252, 121, 224, 306]. Most surveyed users do state a concern for password security and attempt to use better passwords for more security-critical sites but almost universally cite the increasing number of accounts they maintain as preventing them from choosing stronger passwords and not re-using them.

Gaw and Felten sought to quantify this proliferation in a 2006 user study in which users were instructed to log into all web accounts for which they had registered a password. They reported that users had forgotten 25–50% of the accounts for which they had registered and had forgotten passwords to 30% of those accounts they remembered [121]. The best empirical estimates come from a landmark study in 2007 by Flôrencio and Herley that collected large-scale data *in situ* using a browser toolbar [103]. They found the average web user to maintain 25 separate password accounts, with just 6.5 passwords.

2.3 Improvements to passwords

Extensive research has gone into replacing text passwords, perhaps to the detriment of research on passwords themselves [142]. In surveying replacement proposals we find little evidence for any scheme replacing text passwords in the near future, a conclusion reached by several surveys of replacement schemes [227, 249, 45] and essays by senior researchers in the field [114, 143, 142]. Furthermore, most longer-run proposals still rely on secret knowledge at some level.

2.3.1 Cryptographic password-exchange protocols

Cryptography can eliminate the need for passwords to be sent in cleartext from client to server. More broadly, *zero-knowledge proofs of knowledge* allow the prover to conclusively prove knowledge of a password without revealing any information about it. Singh proposed using public-key cryptography to protect password transmission in 1985 [275]. Bellovin and Merrit's 1992 Encrypted Key Exchange (EKE) protocol was the first practical proposal [31], though it required the prover to know the plaintext password. Augmented EKE, a refinement

developed 1993, enables one-way authentication from a client to a server without ever revealing the password to the server [32]. A large number of variants have since been proposed, including protocols provably secure outside the random oracle model [166, 123]. The most practical protocol for the web is the Secure Remote Password (SRP) protocol [318], which is efficient and provides optimal resistance to guessing attacks. Despite being standardised by an RFC for use with TLS [292], SRP has seen little usage at common websites.

2.3.2 Alternate knowledge-based authentication schemes

Random passwords

Assigning randomly generated passwords can provide guaranteed resistance to guessing attacks, though human memory is ill-suited to remembering random strings [38]. Gasser proposed making random strings easier to remember in 1975, developing an algorithm for Multics to generate easy-to-pronounce (and type) strings with a pseudo-natural distributions of vowels and consonants [120]. A close variant was eventually standardised by NIST in 1993 [4], though it was demonstrated to produce a significantly non-uniform distribution [118]. Other proposals for random passwords include Reinhold's Diceware proposal which generates random sequences of dictionary words [248], Spector's Pass-sentence scheme which generates random English sentences [280], King's rebus password scheme which generates images to remember random character strings [172] and Brown's PassMaze protocol which requires users to identify secret words from several lists [56]. None of these proposals has seen significant adoption or large user studies,[9] making them hard to evaluate for practical use.

Passphrases

Requiring multiple words in a password, often called a *passphrase*, may improve security with a similar user experience. Initially, long passwords were not possible because early password hashing routines like `crypt()` had a fixed input space; Porter proposed cipher block-chaining to overcome this problem and allow longer passphrases in 1982 [239]. Passphrases are already deployed in widely used PGP software to protect private keys in case of theft [327]. Usability studies of passphrases [167] have found them to be just as memorable as passwords, subject to an increased rate of typographical errors.[10] Several proposals have been made to reduce the rate of errors for longer passwords, either by storing multiple hashes of a passphrase to recognise entry of nearly correct strings [208, 27] or by providing visual feedback to allow a user to notice typos when they are made [233].

[9] Two separate user studies have found random pronounceable passwords have comparable memorability to self-chosen passwords [328, 57], albeit in a laboratory setting that may not reflect real use.

[10] Passphrases may in fact be easier to type on mobile phones, which have input interfaces optimised for natural language but not for pseudorandom character strings [154].

Mnemonic phrases

In place of complete passphrases, users can condense a sentence like "George Michael and Ann went to the protest on Friday" into a password like `GM&Aw2tpoF`. The original phrase is a mnemonic aid to remember the shorter password. This approach allows faster typing and may offer security similar to the longer phrase if the number of feasible phrases that produce a given password is low. Barton was perhaps the first to propose this idea in 1984, suggesting a variety of mnemonic mappings [28]. Yan et al. found in a study in 2000 that this advice significantly reduced guessability without affecting recall [322]. Kuo et al. performed a dictionary attack on mnemonic-phrase passwords in 2006 [183], finding that many users chose publicly known phrases from quotations, song lyrics etc., though security was significantly better than with text passwords.

Word-association schemes

Given that human recall is greatly enhanced by prompting [38], Smith proposed using pairs of words with a memorable relation in 1987 [277]. In a pilot study, users chose twenty random word pairs and were able to recall the second word correctly 94% of the time when prompted with the first, even after six months of inactivity. Several independent follow-up studies [328, 57, 238] found significantly higher memory rates for such associative passwords than with text passwords [328, 57]. However, 8–25% of spouses were able to guess enough word associations to authenticate. Non-targeted guessing attacks do not appear to have been evaluated formally.

Graphical passwords

Because human memory evolved to remember experiences and emotions and not random strings [38], graphical passwords have been proposed to improve memorability of secrets. User studies suggest graphical schemes are more memorable [55], particularly when multiple secrets must be remembered [66], but can take longer to execute. Biddle et al. provide a good survey of the past decade of research [35], dividing schemes into recall-based, recognition-based, and click-based families.

Recall-based schemes involve a user actively drawing a shape or image from memory. Jermyn et al. introduced the first such scheme, Draw-a-Secret, in 1999 [157]. Among many follow-up proposals, Tao's Pass-Go scheme [290], which limits users to selecting points on a grid, is perhaps the best known due to its eventual adaptation for the Android mobile operating system. Van Oorschot and Thorpe performed successful dictionary-style attacks against recall-based schemes in 2008 [301], exploiting users' tendency to draw simple, symmetrical shapes.

Recognition-based schemes involve a user recognising secret images from a large set of displayed images. Passfaces, proposed by Valentine in 1998, requires users to select pictures of

human faces from a grid of nine possibilities [297]. Dhamija and Perrig's Déjà Vu scheme, proposed in 2000, requires memorising a small random set of images gathered from a large database [87]. A variety of related schemes have been proposed with alternate visual objects to recognise, such as randomly generated inkblots [285]. While the combinatorics of these schemes provides theoretically strong security, human choice appears to greatly limit this. Davis et al. found that user choice in the Passfaces scheme was predictable enough to render it completely insecure [78].

Finally, click-based schemes require memorising a set of points in a prompt image. For example, a user might see a picture of a cat and remember to click on the cat's right ear. In 2005 Wiedenbeck et al. proposed PassPoints, the first scheme based on this principle [312]. Thorpe and van Oorschot found that many users select the same "hot-spots" in a given image, greatly reducing security [294]. To address this problem, in 2008 Chiasson et al. proposed Persuasive Cued ClickPoints [65] which forces users to choose points within a small, random window.

Shoulder-surfing resistant entry

Text and graphical passwords are both vulnerable to physical observation: Tari et al. found that humans can sometimes reconstruct text passwords with a single observation and can usually do so for click-based graphical schemes [291], while Balzarotti et al. demonstrated that text passwords can be deduced with high probability given a video recording of the keyboard [26]. Proposals to resist observation typically require special hardware. Sasamoto et al.'s Undercover scheme [258] uses a special entry box, for example, while Kobara and Imai's LVSVSS method uses physical transparencies to limit the optical angle on which a one-time code can be read [175]. The most promising approach appears to be Kumar et al.'s gaze-based password entry [181], which is backwards-compatible with text passwords and requires only a video camera, which many laptops now have built-in.

Cognitive schemes

Nearly all graphical and multi-word schemes involving static submission of a secret are vulnerable to observation attacks in which an attacker can learn the user's long-term secret. Cryptographically, this problem is easily solved by a challenge-response protocol with the user computing a one-way function $f(\mathbf{x}, \mathbf{c})$ of the long-term secret $\mathbf{x}$ and a random challenge $\mathbf{c}$. However, unless f is simple enough to compute mentally, users remain vulnerable to malware which can observe $\mathbf{x}$ upon entry prior to the computation of f.

Cognitive authentication schemes[11] involve a function f designed to be mentally computable. A simple approach is for the user to hide $\mathbf{x}$ within a sequence of random keystrokes to

[11] The literature is hopelessly inconsistent on terminology for this concept, with alternate terms in use as varied as Leakage-Resilient Passwords and SecHCI.

frustrate naive keyloggers. This was proposed separately by Florêncio and Herley [102] and Cheswick [64] in 2006. Both schemes are easily defeated by more sophisticated malware.

Designing a usable and cryptographically secure scheme has remained an open problem since its formal posing in 1996 by Matsumoto [203]. The most secure schemes, proposed by Hopper and Blum in 2001 [144], can provide cryptographic security by reduction to a known NP-hard problem (learning parity in the presence of noise), but are very difficult to execute and require over three minutes even for skilled users.[12] More intuitive schemes, such as Weinshall's 2006 scheme that involves tracing a path through a two-dimensional grid [308], have usually been quickly cryptanalysed; Weinshall's scheme succumbs to a generic attack using a SAT-solver with only a few observed authentications [124]. GrIDsure [307], another simple scheme requiring only addition, was similarly found to have major security flaws [40]. Coskun and Herley argued from an information theoretic-perspective that the limits of human processing power will prevent cognitive schemes from ever being both usable and secure [73], an argument supported by Yan et al.'s recent meta-study of cognitive proposals and attacks [323].

2.3.3 Non-memory based authentication mechanisms

Physical objects or biometrics enable authentication without memorised secrets. However, the risk of loss or theft typically requires deployment in conjunction with a memory-based scheme in *two-factor authentication*. It is important to study any combination holistically, as Wimberly et al. found that the presence of a second factor can cause humans to pick significantly weaker passwords [315].

Biometrics

Biometrics are any means for identifying humans by difficult-to-forge physiological traits, which may be physical traits such as iris patterns [76], fingerprints [253], or facial structure [192], or behavioural traits such as a user's voice [18], handwriting [25], or typing style [210]. This is a deep research area; Jain et al. provide a good survey of physical biometrics [153] and Yampolskiy et al. of behavioural ones [320]. Most research focuses on supervised authentication and is of limited application to the "unsupervised" environment of web authentication where biometrics are vulnerable to replay attacks such as a high-resolution iris photograph or a replica fingerprint made of gelatin [202]. Preventing replay without supervision is an open problem. Behavioural biometrics can potentially do so through challenge-response protocols, though typically with false-accept rates of at least 10% [320]. Privacy is also a concern when collecting users' inherent physical traits, though privacy-preserving biometric key-generation schemes (analogous to password hashing) exist to prevent verifiers from needing to store biometric signals in intelligible form [24].

[12] Hopper and Blum reported that even after attaching a terminal to a vending machine and offering free soda to colleagues for completing authentications successfully, people preferred to pay for their drinks.

Hardware tokens

Token-based authentication verifies the user's possession of a physical object instead of a memorised secret. The earliest examples are paper-based schemes, first suggested by Lamport in 1981 [184]. Lamport's scheme printed repeated hashes of a master secret onto a paper token, allowing their use as one-time passwords with the verifier storing only the final hash and updating after each authentication. Lamport's protocol was developed by Haller into S/KEY [135] which was eventually standardised as the OTP protocol [134]. Rubin [255] and Kuhn [179] developed later variants which eliminate the risk of theft of the initial secret in Lamport's scheme by generating independent one-time passwords. Paper-based schemes have seen limited use by general-purpose websites, though some banks have deployed a variant called TAN codes with paper sent out via physical mail [313].

Marginally more expensive tokens printed onto transparencies enable visual cryptography. In 1995 Naor and Shamir demonstrated provably secure encryption using transparencies [217], which they later developed into a challenge-response protocol to authenticate humans [216]. The PassWindow scheme [11], a commercial offering using visual cryptography, claims to withstand 20–30 observed authentications [221].

Greater security can be achieved with more expensive electronic tokens. The RSA SecurID product family [148], which uses a tamper-proof module to generate time-varying cryptographic codes, has dominated the market since the 1990s with tens of millions of devices deployed. This scheme requires an expensive token for each verifier, however, which does not scale for the web.[13] Recently, USB tokens have emerged as a competitive approach, such as YubiKey [325], a low-cost device with only one button, or IronKey, a higher-end device capable of launching its own trusted web browser [146]. Future tokens, such as Stajano's recent Pico proposal [282], may be powerful enough to run public-key authentication protocols continually to assert their presence to the verifier. Still, verifying the presence of the token's owner to mitigate theft remains a major challenge.

Personal computer authentication

In place of a separate token, secrets stored on the user's PC can be used as a second factor, although these inherently limit roaming. The TLS protocol enables mutual authentication using public-key client certificates [89], which are supported by all major browsers. The difficulty of generating and installing client certificates has prevented widespread use of TLS for mutual authentication; it is almost-universally only used to authenticate the server to the client. Proposals for more usable management of secrets include Mannan et al.'s ObjPwd [198], which generates a secret from a file selected by the user, and Adida's BeamAuth [16], which stores secrets in a browser bookmark using the fragment tag.

[13] The security of the RSA SecurID system has also been undermined by a major server compromise in 2011 [53] which may have exposed millions of tokens' keys.

These methods are all vulnerable to malware. Stronger authentication is possible using the *trusted platform module* (TPM) now present in most personal computers [131]. The TPM, a tamper-resistant cryptographic co-processor with its own key storage, can be used to securely identify a user's machine as a second factor for authentication [186, 188]. The use of built-in hardware for authentication has been criticised for facilitating vendor lock-in [21]. It also may be unworkable for privacy reasons, as witnessed in the legal battles ensuing from Intel's 1993 proposal to allow remote querying of processor serial numbers [189].

Mobile device authentication

Using mobile phones as a trusted[14] second factor was suggested at least as early as 2004 by Wu et al. [317]. This is more convenient than dedicated tokens, as users already carry mobile phones everywhere. Of many proposed protocols, the simplest might be Mannan and van Oorschot's MP-auth protocol [199] which only trusts the phone to securely encrypt the user's password and requires no storage. Parno et al.'s Phoolproof protocol [230], by contrast, stores a secret key on the phone that is verified during TLS negotiation prior to entering a password. Configuring communication between the mobile device and computer can be a usability challenge; an elegant solution is to use the visual channel [205] as deployed by the commercial Cronto protocol which uses the phone's camera to receive a cryptogram displayed on screen [194].

Mobile devices often have separate communication channels which can reach the verifier, enabling multi-channel protocols [316]. An early example is the 2002 RSAMobile scheme using the SMS channel to send one-time login codes to the user's phone [147]. Google 2-step verification [126] is a recent deployment allowing users to either receive SMS codes or install an application with a secret key.

2.3.4 Single sign-on schemes

Single sign-on (SSO) schemes enable users to use a single password with multiple verifiers via a trusted intermediary. This limits the number of points of failure in the case of re-used passwords. It is also more convenient for users as only one password needs to be remembered, which may also enable users to remember stronger password. Pashalidis and Mitchell provide a survey and taxonomy of different proposals [231]. The key design decisions are using a local or remote proxy and assuming verifier support (true SSO) or not (pseudo-SSO).

Automated password managers

Automated password managers store a user's passwords after the first time they are typed. Typically built in to web browsers or operating systems, this is a local, pseudo-SSO approach.

[14] Smartphones may be decreasingly trustworthy as malware is increasingly targeted at them [100].

The concept dates at least to Apple's KeyChain software for MacOS 8.6 in 1999 and is now implemented on most web browsers. Riley et al.'s 2006 survey found about two-thirds of users employing a browser with automatic password entry [252]. Recent versions of the Mozilla Firefox and Google Chrome browsers support synchronising saved passwords between different machines using a master password [108, 127].

However, mainstream password managers still rely on user-generated passwords and allow password re-use, undermining potential security gains.[15] In light of this problem, in 2005 Halderman et al. proposed automatically deriving domain-specific passwords by hashing a master password with the verifier's domain [133]. A similar approach was developed by Ross et al. in the PwdHash browser extension which also supports roaming access to domain-specific passwords and has been installed by about 100,000 Mozilla Firefox users [254], though this is a tiny fraction of the total user base. Chiasson et al. conducted a usability study of these password managers in 2006 and found the majority of users were unable to use them as intended to generate strong passwords [67].

Proxy login schemes

Password managers typically require per-machine enrolment and software installation. Gabber et al. proposed a scheme called Janus in 1997 to avoid this using a trusted proxy to automatically insert strong passwords into web forms [116]. This remote, pseudo-SSO approach also requires no changes to verifying servers but entrusts all passwords to a third party. Several proposals have enabled authentication to the trusted proxy through an untrusted terminal: the Impostor [232] scheme using cognitive authentication (§2.3.2), KLASSP [102] using obfuscated password entry (§2.3.2) and URSSA [104] using paper-based one-time passwords(§2.3.3). None have seen practical deployment.

Federated authentication

Federated authentication schemes use a trusted server, often called an *identity provider*, to prove a user's identity to verifiers (called *relying parties*) willing to trust the identity provider. In 1978, Needham and Schroeder first proposed a delegated, symmetric-key authentication protocol [220], which developed at MIT in the 1980s into the well-known Kerberos protocol [283] and was eventually standardised in 1993 [176]. Kerberos has been widely deployed for system login within single corporations or universities.

A variety of schemes have arisen for deploying Kerberos on the web using HTTP cookies and redirection, mostly developed for federated authentication within universities [204, 265, 305, 74, 5].[16] The Shibboleth protocol [211] is designed to facilitate interoperation between such systems but has similarly mostly seen deployment only between universities.

[15] LastPass, a commercial offering, generates passwords automatically [185].

[16] Kerberos has been standardised as a cipher suite for use with TLS [207], but this approach is rarely used.

Microsoft Passport [10], launched in 1999, was an early attempt to deploy Kerberos-style authentication to the web with Microsoft acting as a trusted identity provider. Passport received criticism for its centralised nature and several security problems were found with the protocol [177, 228]. After receiving little uptake, it was subsumed by the Windows CardSpace project [34] before being cancelled in 2011. A more successful proprietary protocol is Facebook Connect [145], which has been deployed on hundreds of relying websites since its 2008 launch. Verified by Visa [8], a single sign-on designed to authenticate online payment card transactions, has also seen widespread adoption despite criticism for security shortcomings [214].

In response to the proprietary protocols, several attempts have been made to design open protocols which allow users to choose from multiple identity providers. The best-known efforts are OpenID, launched in 2006 [246], and the related OAuth protocols [136, 245] which have since been merged into the OpenID Connect draft standard [256]. OpenID has been criticised for leaving users vulnerable to phishing [187] and offering poor usability [86] as well as for poor economic incentives that encourage many identity providers but few relying parties or users [286]. Proposed improvements include building privacy-preserving cryptographic protocols on top of OpenID [85] and improving usability through browser support [287].

BrowserID, a competing protocol based on extended email address verification, was launched by Mozilla in 2011 [107]. BrowserID enables supporting browsers to cache signed certificates from identity providers asserting ownership of a claimed email address, thus enabling users to authenticate without revealing the relying party to their identity provider.

2.4 Password cracking

Morris and Thompson published the first password cracking results in 1979 [212], literally trying every entry in the 250,000 word system dictionary in what was probably the first dictionary attack (though they did not use this term directly). Password cracking appears to have emerged as a hobbyist activity through the 1980s [70], leading to the famous 1988 Morris worm, which propagated in part by guessing passwords using a 431-word password dictionary and several rules to modify passwords [268]. The worm's dictionary was not carefully optimised (§8.2.3), with large numbers of musical terms, relatively uncommon names like `hiawatha` and several inexplicable entries like `williamsburwillie`. Still, the worm demonstrated the utility of guessing non-dictionary words. Klein published detailed results on dictionary attack experiments against 14,000 accounts taken from university systems in 1990 [173], using 24 small dictionaries of a few thousand words each from disparate sources like sports team names and movie titles.

Password cracking evolved rapidly in the years after these studies. Freely available cracking tools emerged like Alec Muffett's Crack [213], which introduced *mangling* rules to turn a single dictionary password like `joe` into variants like `Joe`, `J0E` and `eoj`. Today John the Ripper [1] is the dominant open-source cracking tool in addition to several commercial offerings.

year	study	hash	platform	speed (MHz)	cost ($\frac{\text{US\$}}{\text{MHz}}$)
1991	Leong & Tham [190]	DES	PC	0.02	—
1991	Leong & Tham [190]	DES	hardware	4.2	$786.42
1998	Gilmore et al. [122]	DES	hardware	92,666	$3.75
2006	Mentens et al. [209]	DES	FPGA	10,000	$0.84
2011	Sprengers [281]	MD5	GPU	880	$0.39

Table 2.1: Password cracking speed through the years. Speed results are reported in millions of base hashes per second (MHz). The rate of a real password attack will be slower due to the use of iterated hashing (§2.2.1). Cost figures are converted from authors' contemporary estimates into 2012 US dollars using US Bureau of Labor Statistics inflation figures [226] and the XE Universal Currency Converter [149]. Note that costs only reflect capital. Energy costs are typically not reported making it difficult to compare the total cost of brute-force attacks.

Narayanan and Shamitkov were perhaps the first to formalise mangling rules in 2005 [218], modeling password construction as a Markov process in which each character is a probabilistic function of the previous n characters. Weir et al. proposed using a probabilistic context-free grammar to generate mangling rules automatically from a training set in 2009 [310]. Both efforts can be viewed as attempts to approximate the distribution of passwords using a model distribution, a concept we will return to in §5.6. Weir's probabilistic context-free grammar approach has out-performed the Markov model approach in head-to-head comparisons and is now favoured by the password-cracking community [200, 168].

2.4.1 Optimising brute-force performance

Since password-cracking libraries with mangling rules can generate an effectively infinite sequence of possible passwords, their effectiveness depends on the rate at which guessed passwords can be evaluated, measured in guesses per second (Hz). Morris and Thompson considered the 800 Hz rate that was possible to achieve in software against the original UNIX hashing algorithm[17] unacceptable and aimed to increase computational cost when designing `crypt()` by tweaking the DES algorithm to prevent hardware-accelerated search [212].

A decade later, Feldmeir and Karn were able to top 1 kHz against `crypt()` with an optimised software implementation [99]. They warned that the "`crypt()`/sec/dollar" ratio had increased by five orders of magnitude since `crypt()` was introduced and that within a decade it might be feasible to exhaust the entire 56-bit input space. A year later, Leong and Tham proposed a theoretical 160 kHz hardware design able to build a precomputed dictionary of all passwords in one year for just US$2,000 [190]. Feldmeir and Karn's ten-year prediction proved highly

[17] The original UNIX hash was based on the US Army's World War II-era M-209 mechanical cipher machine.

accurate, as DES was publicly brute-forced in 1998 [122].[18]

Recent work has utilised reconfigurable hardware for lower-cost password cracking. The 2006 COPACABANA project demonstrated the cost-effectiveness of using FPGAs for cryptographic brute-force [182]. Mentens et al. studied password cracking specifically on FPGAs [209]. Password cracking has also been implemented on cheaper general-purpose hardware like the Sony Playstation gaming console [174] and high-end graphics processors [281]. Table 2.1 enumerates the efficiency and cost of various password cracking platforms; Sprengers' recent results on graphical processors are the most cost-effective currently known.

2.4.2 Precomputed dictionaries and rainbow tables

Brute-force attacks are significantly more useful if precomputed values can be shared by cooperating attackers.[19] Morris and Thompson warned of the possibility of storing the hashes of common passwords in an *encrypted dictionary attack* [212]. This approach is far more appealing using the time-memory trade-off pioneered for cryptographic key search by Hellman in 1980 [138] in which only the end-points of long chains of password hashes are stored, greatly reducing storage requirements.

Oechslin introduced a refinement of this approach in 2003 called *rainbow tables* [225] which again greatly improve efficiency. Rainbow tables can be computed for a subset of possible passwords such as all 8-character strings or can use a probabilistic reduction function to focus on likely passwords [218]. The RainbowCrack project has collaboratively constructed rainbow tables of all 9-character strings for the MD5 and LM hash algorithms, which each took $\approx 2^{56}$ computation to construct and enable nearly any 9-character password to be broken with $\approx 2^{37}$ hash iterations using a $\approx$ 900 GB lookup table.

2.5 Evaluating guessing difficulty

It has long been of interest to analyse how secure a given data set of passwords is against guessing attacks, dating at least to Morris and Thompson's seminal 1979 analysis of 3,000 passwords [212]. They recovered 84% of available passwords by trying all 6-character ASCII strings, variations of available usernames and every entry in the 250,000-word system dictionary. Unfortunately, they reported the results of all approaches mixed together, while noting that the dictionary approach was more efficient. They also reported some basic statistics such as password lengths (71% were six characters or fewer) and frequency of non-alphanumeric characters (14% of passwords). These two approaches, password cracking and semantic evaluation, have been the basis for many studies in the thirty years since.

[18] While `crypt()` uses $25 \approx 2^{4.6}$ iterations of DES, only $95^8 \approx 2^{52.5}$ strings of 8 printable characters are usually searched, meaning an exhaustive search of `crypt()` requires $\approx 2^{57.1}$ work, twice as much as basic DES.

[19] Note that precomputed attacks are only possible if passwords are hashed without salting (§2.2.1).

2.5.1 Cracking evaluation

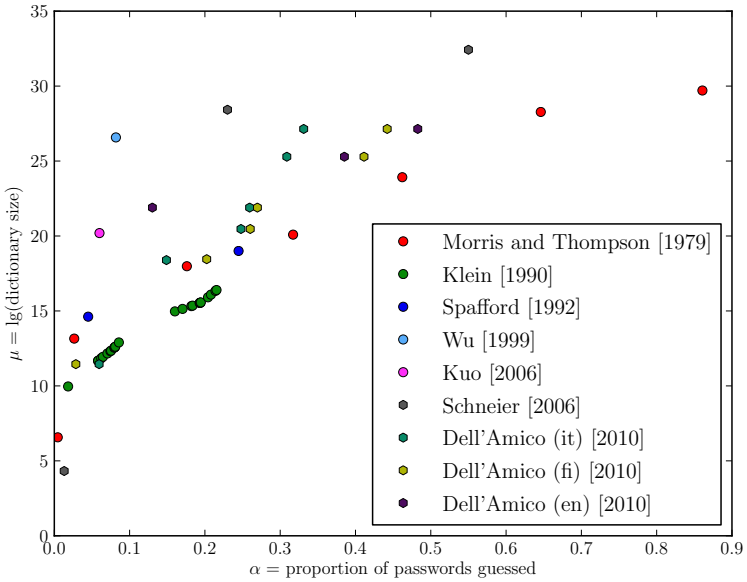
Many researchers have simulated password cracking attacks against available password data sets to evaluate password strength. Klein published the first detailed results on the effectiveness of 24 small dictionaries composed of different types of words in 1990 [173]. Significant variation in efficiency was observed, although Klein noted a trade-off between efficiency and dictionary size with the most efficient lists like surnames and sports teams typically being very short. Spafford performed a similar experiment in 1992 [279] using a larger dictionary and found similar crack times, as seen in in Figure 2.1a.

Subsequent research has typically utilised pre-existing password cracking libraries and reported much less detail on cracking strategy. For example, Wu studied password cracking against Kerberos tickets in 1999 and was able to crack 8.8% using a dictionary of over 100 M words [319]. Kuo et al. studied mnemonic passwords in 2006 and, as a by-product, broke 11% of passwords collected from volunteer subjects with a dictionary of 1.1 M words (also breaking 4% of mnemonic passwords) [183].

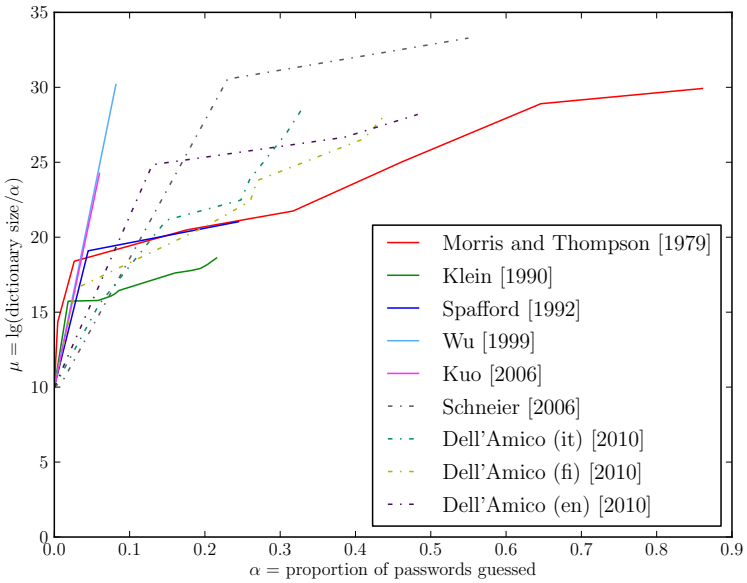
Recently, large-scale password leaks from compromised websites have provided a new source of data on cracking difficulty. For example, Schneier analysed about 50,000 passwords obtained via phishing from the social network MySpace in 2006 [267]. A more in-depth study was conducted by Dell'Amico et al., who studied the MySpace passwords as well as those of two other websites using a large variety of different dictionaries [81]. Weir et al. used the 2009 leak of 32 million RockYou passwords to examine the effect of password-composition rules on cracking efficiency [309]. Suprisingly, these studies all found lower cracking efficiency than previous studies of system passwords, which users were presumably more motivated to choose securely.

Overall, reported numbers on password cracking efficiency vary substantially between different studies, as shown in Figure 2.1. All password-cracking studies have found strongly diminishing efficiency as more passwords are guessed, as shown in Figure 2.1b, which is likely a fundamental property of the underlying distribution. There is plentiful data on dictionaries needed to break 25–50% of accounts, which usually requires a dictionary size of 2^{20}–2^{30} passwords. There is less data on the efficiency of very small dictionaries, which usually are not reported separately. There is even less data for much larger dictionaries—only[20] Morris and Thompson were able to break more than 50% of available passwords, and their results cannot be assumed to apply directly to modern password choices.

[20] A 2007 study by Cazier and Medlin claimed to break 99% of passwords taken from a real e-commerce website, but few details were provided [63].



(a) Historical password cracking efficiency, raw dictionary size



(b) Historical password cracking efficiency, equivalent dictionary size

Figure 2.1: Historical results in password cracking efficiency. Circles and solid lines represent system passwords, squares and dashed lines represent web passwords. In Figure 2.1a, the increasing size of dictionaries is plotted logarithmically against the success rate achieved. In Figure 2.1b, the dictionary sizes are adjusted to incorporate the inherent need for more guesses to crack more passwords. This concept will be important for developing statistical metrics of guessing difficulty in §3.2.4.

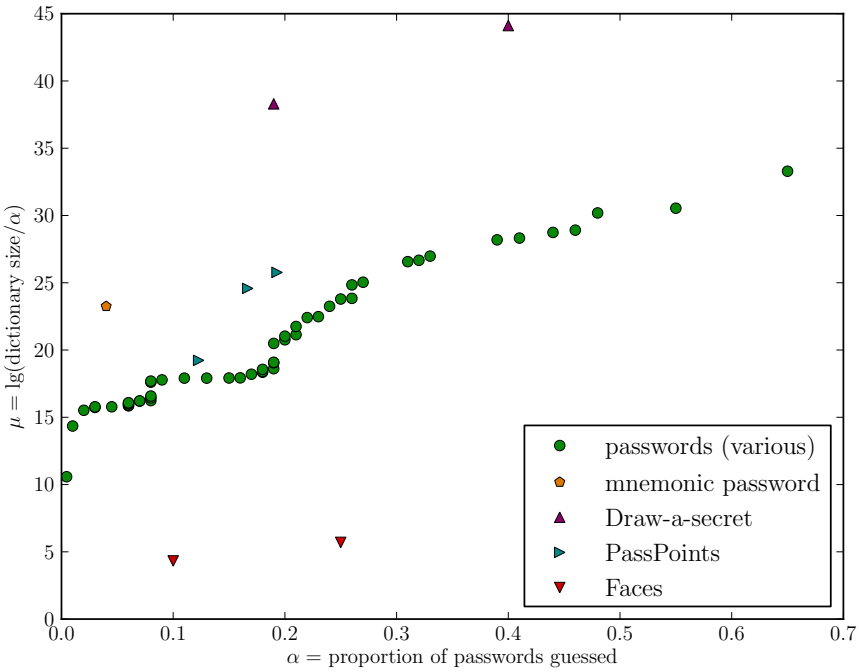


Figure 2.2: Comparison of attacks on graphical and mnemonic password schemes against traditional password cracking. Except for the badly broken Faces scheme, the attacks are less efficient against alternative schemes than against traditional password schemes, although this may represent relative lack of research on guessing against graphical schemes.

Guessing attacks on other authentication systems

A small number of academic studies have performed cracking-style attacks on other schemes using human-chosen secrets (§2.3.2). Kuo et al. used a dictionary of 400k likely phrases to recover 3% of user-created mnemonic-phrase passwords [183]. Davis et al. attacked a Passfaces-style scheme which they called "Faces" in 2004 [78] and found that user preference for attractive faces was prevalent enough that they could break 10% of accounts with just two guesses. Thorpe et al. attacked the click-based PassPoints scheme in 2007 [294] and found that users exhibited a strong tendency to select a small number of prominent points, enough to break nearly 20% of accounts with 2^{23} guesses. In 2008 van Oorschot and Thorpe developed dictionary attacks against human-drawn passwords [301], though these were not nearly as efficient, requiring around 2^{36} guesses to break 19% of accounts.

These guessing results are compared to those achieved for passwords in Figure 2.2. While graphical schemes appear comparatively better against guessing attacks, caution is needed as considerably fewer attacks have been developed. For each scheme proposed, there has only been a single academic proof-of-concept attack, whereas password cracking has seen dozens of publications as well as considerable interest from outside the academic community.

year	study	length	% digits	% special
1989	Riddle et al. [251]	4.4	3.5	—
1990	Sawyer et al. [260]	5.7	19.2	0.7
1992	Spafford [279]	6.8	31.7	14.8
1999	Wu [319]	7.5	25.7	4.1
2004	Campbell and Bryant [61]	8.0	78.0	3.0
2006	Cazier and Medlin [63]	7.4	35.0	1.3
2009	RockYou leak [84]	7.9	54.0	3.7

Table 2.2: Commonly estimated attributes of passwords: length, percentage containing digits, and percentage containing special (non-alphanumeric) characters.

2.5.2 Semantic evaluation

There have been many studies on the semantics of passwords, a topic which has interested psychologists and linguists as well as computer security researchers. Riddle et al. performed linguistic analysis of 6,226 passwords in 1989, classifying them into categories such as names, dictionary words, or seemingly random strings [251]. Cazier et al. repeated this process in 2006 and found that hard-to-classify passwords were also the hardest to crack [63].

Many studies have reported simpler data points like average length and types of characters used, as summarized in Table 2.2. The estimates vary so widely that it is difficult to infer much of use for security engineers. The main trends are a tendency towards six to eight characters of length and a strong dislike of non-alphanumeric characters in passwords.[21]

More formal attempts to study password structure have been conducted to develop proactive password checking algorithms (§2.2.4), such as Bergadano et al.'s use of a decision tree to classify different types of passwords [33]. This research can be seen as the the dual of creating mangling rules for cracking (§2.4). For example, Markov models were used as early as 1993 for proactive password checking [77] before being adapted for password cracking [218, 200] and have since been re-adapted for password checking [62].

Estimating entropy from password structure

Many studies attempt to assess the strength of an individual password by examining its structure. Typically this requires modeling the process by which the password was generated, for example "five letters followed by two digits and a punctuation character" and then calculating the total space of passwords which this process could produce. This approach was used as

[21] It is often suggested that users avoid characters which require multiple keys to type, but this has not been formally established.

early as 1972 by Anderson [20]. The size of this space can be used to estimate the strength of a password, often referred to imprecisely as "entropy" after taking a logarithm.[22]

It is unclear when the term entropy caught on in relation to password analysis. It is absent from most early literature and the earliest identifiable usage seems to be 1999 [319]. The term was cemented by the 2006 FIPS Electronic Authentication Guideline [58], which provided a "rough rule of thumb" formula for estimating entropy from password characteristics. This standard has since been used in several password studies with too few samples to compute statistics on the entire distribution [103, 273, 169, 168]. While the FIPS algorithm is somewhat ad hoc, more systematic formulas have been proposed, such as that of Yan [321] or Shay et al. [273], which add the apparent entropy from many different elements of a password's structure.

2.5.3 Problems with previous approaches

Three decades of work on password guessing has produced sophisticated cracking tools and many disparate data points, but several methodological problems remain:

Comparability

Authors rarely report cracking results in a format which is straightforward to compare with previous benchmarks. To our knowledge, Figure 2.1 is the first comparison of different data points of dictionary size and success rate, though direct comparison is difficult since authors all report efficiency rates for different dictionary sizes. Password cracking tools only loosely attempt to guess passwords in decreasing order of likeliness, introducing imprecision into reported dictionary sizes. Worse, some studies report the running time of cracking software instead of dictionary size [63, 322, 284], making comparison difficult.

Repeatability

Precisely reproducing password cracking results is difficult. Early password-cracking studies relied on custom dictionaries which are presumably lost to history. Modern password-cracking software relies not only on large dictionaries but the precise configuration of code to modify them. John the Ripper [1], used in most publications of the past decade, has been released in 21 different versions since 2001 and makes available for use 20 separate word lists, along with proprietary word lists and many configuration options. Other studies have used proprietary password-cracking software which isn't available to the research community [63, 267]. Thus nearly all studies use dictionaries varying in both content and ordering, making it difficult

[22] This terminology is mathematically incorrect because entropy (§§3.1.1–3.1.2) measures a complete probability distribution, not a single event (one password). The correct metric for a single event is *self-information* (or *surprisal*). This is perhaps disfavoured because it is counter-intuitive: passwords should avoid including information like names or addresses, so high-information passwords sound weak.

to compare the resistance of a new data set to the exact cracking attack used in previously published results.

Lack of research precision in password cracking software

Password-cracking programs produce poor output for evaluating their own efficiency. For comparative purposes, it is most interesting to see the total number of passwords attempted and the fraction of available accounts broken. Most password cracking software does not output this or strictly attempt to guess passwords in increasing likelihood, which may be inefficient or even impossible when complicated mangling rules are being applied. Kelley et al., for example, needed to reverse-engineer large parts of the cracking library they used in order to identify the order in which individual passwords would be guessed [168]. This is a particular problem when evaluating very small dictionaries for which cracking libraries are usually not optimised.

Evaluator dependency

Password-cracking results are inherently dependent on the appropriateness of the dictionary and mangling rules for the data set under study. Thus it is difficult to separate the effects of more-carefully chosen passwords from the use of a less-appropriate dictionary. This is particularly challenging in data-slicing experiments [309, 168], which require simulating an equally good dictionary attack against each subpopulation.

Unsoundness of entropy

Entropy is not mathematically appropriate as a measure of guessing resistance, as we will demonstrate in §3.2. It is also very difficult to estimate from a sample (§5.2). Approximation formulas can be used with small sample sizes but inherently make many assumptions about password selection. In practice, entropy estimates have performed poorly as predictors of empirical cracking difficulty [309, 168], leading to password cracking being favoured despite being less precise and harder to repeat.

Don't keep a man guessing too long—
he's sure to find the answer somewhere else.

—Mae West, 1933

Chapter 3

Metrics for guessing difficulty

We now turn our attention to the problem of measuring the guessing difficulty of a distribution of human-chosen secrets, the core technical contribution of this dissertation. To overcome the practical problems of password cracking and semantic evaluation, we model password selection[1] as a random draw $X \xleftarrow{\mathrm{R}} \mathcal{X}$ from an *underlying distribution* $\mathcal{X}$. For now, we assume that $\mathcal{X}$ is completely known to the attacker.[2] For an adversary given a (possibly singleton) set of unknown values $\{X_1, X_2, \ldots X_k\}$ and access to an oracle for queries of the form[3] "is $X_i = x$?" we consider several metrics to evaluate how efficiently our adversary can correctly identify some proportion of the unknown X_i. Our goal in this chapter is to develop sound metrics; we will analyse real data in subsequent chapters.

3.1 Traditional metrics

3.1.1 Shannon entropy

Shannon entropy H_1 (or simply H), introduced by Claude Shannon in 1948 [271], is the best-known measure of the "uncertainty" of a value drawn from $\mathcal{X}$:

$$H_1(\mathcal{X}) = \sum_{i=1}^{N} -p_i \lg p_i \tag{3.1}$$

Shannon developed the metric while studying information loss on analog phone lines. He proved it is the only definition which is continuous (small changes to the probability distribution lead to small changes in measured uncertainty), maximised by a uniform distribution and

[1] Our mathematical notation is introduced and defined in §1.4.

[2] We will address problems due to incomplete knowledge of $\mathcal{X}$ in §5, and due to training on a distinct distribution $\mathcal{Y} \neq \mathcal{X}$ in §8.

[3] Note that this formulation assumes password hashes, if available, are salted. Otherwise an attacker could efficiently ask "is *any* $X = x$?"

additive for joint distributions. Perhaps the most important application is Shannon's source coding theorem [271], which states that $X \xleftarrow{\mathrm{R}} \mathcal{X}$ can be encoded using a Huffman code with average bit length $\leq H_1(\mathcal{X}) + 1$.

An easy corrollary is that the average number of guesses needed to identify X if an attacker can ask "Is $X \in \mathcal{S}$?" for arbitrary subsets $\mathcal{S} \subseteq \mathcal{X}$ is $H_1(\mathcal{X})$ because the attacker can query one bit of the optimal encoding at a time. For an attacker who must guess possible values individually, Shannon entropy has no direct correlation to guessing difficulty.[4]

3.1.2 Rényi entropy and its variants

Rényi entropy H_n is a generalisation of Shannon entropy developed by Alfred Rényi in 1961 [250]. It is parameterised[5] by a real number $n \geq 0$:

$$H_n(\mathcal{X}) = \frac{1}{1-n} \lg \left(\sum_{i=1}^{N} p_i^n \right) \tag{3.2}$$

Rényi proved that this formula defines all possible uncertainty measures which meet Shannon's axioms if the additivity property is relaxed.[6] In the limit as $n \to 1$, Rényi can be shown using l'Hôpital's rule to converge to the classic definition of Shannon entropy, hence Shannon entropy is denoted H_1. H_n is a monotonically decreasing function of n; it is equal to $\lg N$ for all n for a uniform distribution and is weakly decreasing with n for all non-uniform distributions.

We are interested in several special cases:

Hartley entropy H_0

For $n = 0$, Rényi entropy becomes:

$$H_0 = \lg N \tag{3.3}$$

First used by Ralph Hartley two decades prior to Shannon entropy [137], H_0 measures only the size of a distribution and ignores the probabilities.

Collision entropy H_2

For $n = 2$, Rényi entropy becomes:

$$H_2 = -\lg \left(\sum_{i=1}^{N} {p_i}^2 \right) \tag{3.4}$$

[4] H_1 has further been claimed to poorly predict actual password-cracking times [309, 168], though these studies used naive estimates of H_1 which aren't accurate (§5.1).

[5] Rényi entropy is usually denoted H_α; we write H_n to avoid confusion with α as a desired success rate.

[6] The weaker additivity property of Rényi entropy requires only that $H_n(\mathcal{XY}) = H_n(\mathcal{X}) + H_n(\mathcal{Y})$ if $\mathcal{X}$ and $\mathcal{Y}$ are independent. H_1 further admits a definition of *conditional entropy* where $H_n(\mathcal{XY}) = H_1(\mathcal{X}) + H_n(\mathcal{Y}|\mathcal{X})$.

This value is called the *collision entropy* because it is equal to $-\lg P(X = Y)$, the probability that two independently chosen values $X \xleftarrow{R} \mathcal{X}$, $Y \xleftarrow{R} \mathcal{X}$ will be equal.

This models the success of an adversary who, instead of guessing in decreasing order of probability, randomly selects guesses $x \xleftarrow{R} \mathcal{X}$. This strategy could arise if an attacker is trying evade an intrusion detection system which detects if the pattern of guesses is skewed compared to the underlying password distribution.

Min-entropy H_∞

As $n \to \infty$, Rényi entropy becomes:

$$H_\infty = -\lg p_1 \tag{3.5}$$

This metric is only influenced by the probability of the most likely event in the distribution. Min-entropy is important as an asymptotic limit on the number of uniformly random bits which can be deterministically *extracted* from a distribution [222].[7] In this sense it can be interpreted as the minimum amount of randomness contained in a distribution.

It similarly functions as a worst-case security metric for guessing attacks, measuring security against an attacker who only guesses the most likely item before giving up. Min-entropy is a lower bound not only for all other Rényi entropies, but for all of the guessing metrics we will define, making it useful for conservative security analysis.

3.1.3 Guesswork

If an adversary must guess only one value at a time, none of the above uncertainty metrics directly relate to guessing difficulty. A more applicable metric is the expected number of guesses required to find X if the attacker proceeds in optimal order:

$$G_1(\mathcal{X}) = E\left[\#_{\text{guesses}}(X \xleftarrow{R} \mathcal{X})\right] = \sum_{i=1}^{N} p_i \cdot i \tag{3.6}$$

This measure was studied as simply G by Massey [201], it was later independently named "guessing entropy" by Cachin [60] and "guesswork" by Pliam [236]. We prefer the latter term to stress that this measure is not closely related to the Rényi family of entropy measures and not logarithmically scaled.

While G_1 soundly measures security against an adversary willing to exhaustively guess all values from $\mathcal{X}$, this formulation is overly conservative and can produce absurd results. For

[7] Cachin [60] proved that in a relaxed model, the number of almost-uniform bits which can be probabilistically extracted from a distribution is the Rényi entropy H_n for some $1 < n < 2$, a value called *smooth entropy*.

example, in the RockYou data set (§D) at least twenty users (more than 1 in 2^{21}) use apparently pseudorandom 32-character hexadecimal strings as passwords. These ensure a lower bound on the overall guesswork using the following lemma, proved in §B.1:

$$G_1(p \cdot \mathcal{X} + q \cdot \mathcal{Y}) \geq p \cdot G_1(\mathcal{X}) + q \cdot G_1(\mathcal{Y}) \tag{3.7}$$

Assuming the random-looking passwords are drawn from $\mathcal{U}_{2^{128}}$, for which $G_1(\mathcal{U}_{2^{128}}) > 2^{127}$, we can thus conclude that $G_1(\text{RockYou}) \geq 2^{-21} \cdot 2^{127} = 2^{106}$, no matter how the other passwords are chosen! Thus G_1 provides little insight into practical attacks and we will later show it is difficult to estimate from sampled data (§5.2).

3.2 Partial guessing metrics

Traditional metrics fail to model the ability of real-world attackers to cease guessing against the most difficult accounts. As discussed in §2.5.1, cracking evaluations explicitly look for weak subspaces of passwords to attack and typically report the fraction of accounts broken. Having many available accounts enables a *partial guessing* attack which trades a lower proportion of accounts broken for increased guessing efficiency.

Formally, if Eve must sequentially guess each of k values drawn from $\mathcal{X}$, she will need $\sim k \cdot G_1(\mathcal{X})$ guesses on average. However, a second guesser Mallory needing to break only $\ell < k$ of the values can guess the most likely password for all k accounts, then the second-most likely value and so on until ℓ have been broken. As ℓ decreases, Mallory's efficiency increases as the attack can omit progressively more low-probability values. For large values of k and ℓ, Mallory will only need to guess the most popular β values such that $\sum_{i=1}^{\beta} p_i \geq \alpha$, where $\alpha = \frac{\ell}{k}$. There are several metrics for partial guessing attacks:

3.2.1 β-success-rate

A very simple metric, first formally defined by Boztaş [50], measures the expected success for an attacker limited to β guesses per account:

$$\lambda_\beta(\mathcal{X}) = \sum_{i=1}^{\beta} p_i \tag{3.8}$$

3.2.2 α-work-factor

A related metric, first formalised by Pliam [236], evaluates the fixed number of guesses per account needed to break a desired proportion α of accounts.

$$\mu_\alpha(\mathcal{X}) = \min\left\{ \mu \middle| \sum_{i=1}^{\mu} p_i \geq \alpha \right\} \tag{3.9}$$

If $\mu_\alpha(\mathcal{X}) = n$, an attacker must use a dictionary of size $\mu \geq n$ to break an individual account with probability α, or equivalently to break an expected fraction α of many accounts.

3.2.3 α-guesswork

While λ_β and μ_α are closer to measuring real guessing attacks, both ignore the fact that a real attacker can stop early after successful guesses. While breaking a fraction α of accounts requires up to μ_α guesses per account, some will require fewer guesses. We introduce a new metric to reflect the expected number of guesses per account to achieve a success rate α.

A partial guessing attack will require only 1 guess at a proportion p_1 of values which take on the most likely value, 2 guesses for the proportion p_2 which take on the second most likely value and so on, leading to a summation similar to Equation 3.6. Against all values not in the dictionary the attacker will make μ_α guesses, giving a total expected number of guesses:

$$G_\alpha(\mathcal{X}) = (1 - \lceil\!\lceil \alpha \rceil\!\rceil) \cdot \mu_\alpha(\mathcal{X}) + \sum_{i=1}^{\mu_\alpha(\mathcal{X})} p_i \cdot i \tag{3.10}$$

The traditional guesswork metric G_1 (or just G) is a special case of this metric with $\alpha = 1$, representing an attacker determined to break all accounts. We could equivalently define G_β for an attacker limited to β guesses, but this is less useful as for small β the effect of stopping early is negligible.

Note that our definition uses the expression $(1 - \lceil\!\lceil \alpha \rceil\!\rceil)$ to round down the proportion of unsuccessfully guessed values and not $(1 - \alpha)$. This is because an attacker making μ_α guesses may actually succeed with probability greater than α. For example, for any $0 < \alpha \leq p_1$, an attacker making $\mu_\alpha = 1$ guess will succeed with probability $\lceil\!\lceil \alpha \rceil\!\rceil = p_1$. We can define this rounding operation succinctly using our β-success-rate definition as:

$$\lceil\!\lceil \alpha \rceil\!\rceil = \lambda_{\mu_\alpha(\mathcal{X})}(\mathcal{X}) = \sum_{i=1}^{\mu_\alpha(\mathcal{X})} p_i \tag{3.11}$$

3.2.4 Effective key-length conversion

While λ_β, μ_α and G_α are not measures of entropy, it is convenient to convert them into units of bits. This enables direct comparison of all metrics as a logarithmically scaled attacker workload which is intuitive to programmers and cryptographers. This can be thought of as an "effective key-length" as it represents the size of a randomly chosen cryptographic key which would give equivalent security.[8]

[8] Boztaş used the term effective key-length specifically to refer to $\tilde{\mu}_{0.5}$ [50]. O'Gorman also used the term to denote the equivalent false positive and false negative rates in biometrics [227].

A metric's effective key-length is the (logarithmic) size of a uniform distribution $\mathcal{U}_N$ which has the same value of the guessing metric. This approach is often used implicitly in cryptography, where skewed human-chosen distributions are abstracted as a uniform distribution of size H_∞ [112, 123]. This approach extends easily to partial guessing metrics.

For β-success-rate, since we have $\lambda_\beta(\mathcal{U}_N) = \frac{\beta}{N}$ we say that any distribution $\mathcal{X}$ is equivalent with respect to λ_β to a uniform distribution of size $N = \frac{\beta}{\lambda_\beta(\mathcal{X})}$. We take a logarithm to produce an effective key-length metric $\tilde{\lambda}_\beta$, using a tilde to denote the conversion to bits:

$$\tilde{\lambda}_\beta(\mathcal{X}) = \lg\left(\frac{\beta}{\lambda_\beta(\mathcal{X})}\right) \tag{3.12}$$

The conversion formula for α-work-factor is very similar:

$$\tilde{\mu}_\alpha(\mathcal{X}) = \lg\left(\frac{\mu_\alpha(\mathcal{X})}{\lceil\alpha\rceil}\right) \tag{3.13}$$

We again use $\lceil\alpha\rceil$ (as defined in Equation 3.11) in place of α in the denominator because μ_α increases as a step function as α increases. Without this correction, $\tilde{\mu}_\alpha$ would decrease over each interval of α where μ_α is constant, giving a misleading over-estimate of security as seen in Figure 3.1. Using $\lceil\alpha\rceil$ ensures that $\tilde{\mu}_\alpha$ is monotonically increasing and avoids the undesirable possibility that $\lim_{\alpha\to 0^+} \tilde{\mu}_\alpha(\mathcal{X}) = \infty$.

To convert G_α, we consider that an attacker desiring to break a proportion α of accounts will average G_α guesses per account and break a proportion $\lceil\alpha\rceil$ of them, giving an overall rate of one successful guess per $\frac{G_\alpha}{\lceil\alpha\rceil}$ guesses. Against the uniform distribution $\mathcal{U}_N$, an attacker will break an account every $\frac{N+1}{2}$ guesses, giving us the possible formula:

$$\tilde{G}^?_\alpha(\mathcal{X}) = \lg\left[\frac{2\cdot G_\alpha(\mathcal{X})}{\lceil\alpha\rceil} - 1\right] \tag{3.14}$$

This definition, however, isn't constant for a uniform distribution $\mathcal{U}_N$:

$$\begin{aligned}
\tilde{G}^?_\alpha(\mathcal{U}_N) &= \lg\left[\frac{2\cdot G_\alpha(\mathcal{U}_N)}{\lceil\alpha\rceil} - 1\right] \\
&= \lg\left[\frac{2}{\lceil\alpha\rceil}\cdot\left((1-\lceil\alpha\rceil)\cdot\mu_\alpha(\mathcal{U}_N) + \sum_{i=1}^{\mu_\alpha(\mathcal{U}_N)} p_i\cdot i\right) - 1\right] \\
&= \lg\left[\frac{2}{\lceil\alpha\rceil}\cdot\left((1-\lceil\alpha\rceil)\cdot\lceil\alpha\rceil N + \frac{1}{N}\sum_{i=1}^{\lceil\alpha\rceil N} i\right) - 1\right] \\
&= \lg\left[\frac{2}{\lceil\alpha\rceil}\cdot\left(\lceil\alpha\rceil N - \lceil\alpha\rceil^2 N + \frac{\lceil\alpha\rceil^2 N + \lceil\alpha\rceil}{2}\right) - 1\right] \\
&= \lg\left[(2-\lceil\alpha\rceil)N\right]
\end{aligned} \tag{3.15}$$

Note that we have $\tilde{G}^?_1(\mathcal{U}_N) = \lg N$, as desired, but $\tilde{G}^?_\alpha(\mathcal{U}_N) \to \lg 2N = \lg N + 1$ as $\alpha \to 0$. This reflects a subtle property of guessing: for a uniform distribution, repeatedly trying guesses for

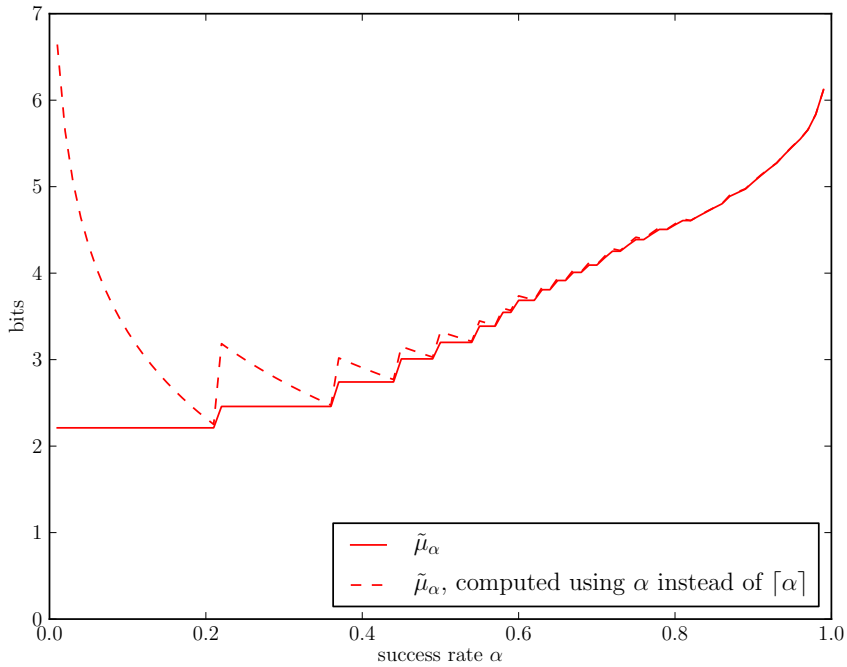


Figure 3.1: This figure shows the importance of using $\lceil\!\lceil\alpha\rceil\!\rceil$ in place of α in the definition of $\tilde{\mu}_\alpha$ in Equation 3.13. This plot shows $\tilde{\mu}_\alpha$ for the distribution of surnames in South Korea (§7.2.1), which is highly skewed with $p_1 \approx 0.22$ for the surname 'Kim'. If α is used, a misleading sawtooth pattern emerges represented by the dashed line where security appears to increase for lower α, with $\tilde{\mu}_\alpha \to \infty$ as $\alpha \to 0$.

the same unknown value will require an expected $\frac{N+1}{2}$ guesses per success. Trying a single value for each account requires N guesses per success. This occurs because each unsuccessful guess decreases uncertainty and makes future guessing more efficient.

Still, it is best to define the property to make $\tilde{G}_\alpha(\mathcal{U}_N) = \lg N$ for all α and thus make it easier to compare to our other metrics. We can see from the last line of Equation 3.15 that this can be done by subtracting a simple correction factor:

$$\tilde{G}_\alpha(\mathcal{X}) = \lg\left[\frac{2 \cdot G_\alpha(\mathcal{X})}{\lceil\!\lceil\alpha\rceil\!\rceil} - 1\right] - \lg(2 - \lceil\!\lceil\alpha\rceil\!\rceil) \tag{3.16}$$

This correction factor is a continuous, monotonically decreasing function of α, ranging from $\lg 2 = 1$ as $\alpha \to 0$ to $\lg 1 = 0$ as $\alpha \to 1$. The usefulness of this correction is seen visibly in Figure 3.2, where it ensures that $\tilde{\mu}_\alpha \sim \tilde{G}_\alpha$ for low α, which matches our intuition that the benefit of being able to stop early is neglible for attacks with a low desired success rate α.

Unit conversion by change of logarithm base

Shannon entropy can be converted into alternative units by changing the base of the logarithm used from the traditional base of 2. Standard alternatives include base 10, giving units called

dits (also called *hartleys* or *bans*), and the natural logarithm (using Euler's constant e as the base), giving units of *nats* (also called *nits* or *nepits*). We can achieve the same unit conversion simply by changing the logarithm base in Equations 3.12, 3.13 and 3.16.

Converting to dits can be useful if we are studying PINs, which are canonically represented as decimal digits. In Figure 3.2, the right y-axis displays units of dits. With the conversion, it is easy to see that $\mathcal{U}_{10^3}$, equivalent to a random 3-digit PIN, provides exactly 3 dits of security. Simple re-labeling of the axis is possible because the conversion from base 2 to base b is equivalent to a multiplication by $\frac{1}{\log_2(b)}$; 1 dit is equal to $\frac{1}{\log_2(10)} = 3.32$ bits.

A larger base could be used to measure text passwords in units of equivalent random "characters". Unlike digits, however, there are many ways to count the number of possible characters: 26 English letters, 52 letters including case, 62 alphanumeric characters, 96 printable ASCII characters, or 110,181 characters in the recent Unicode 6.1 standard [13].

3.2.5 Guessing curves

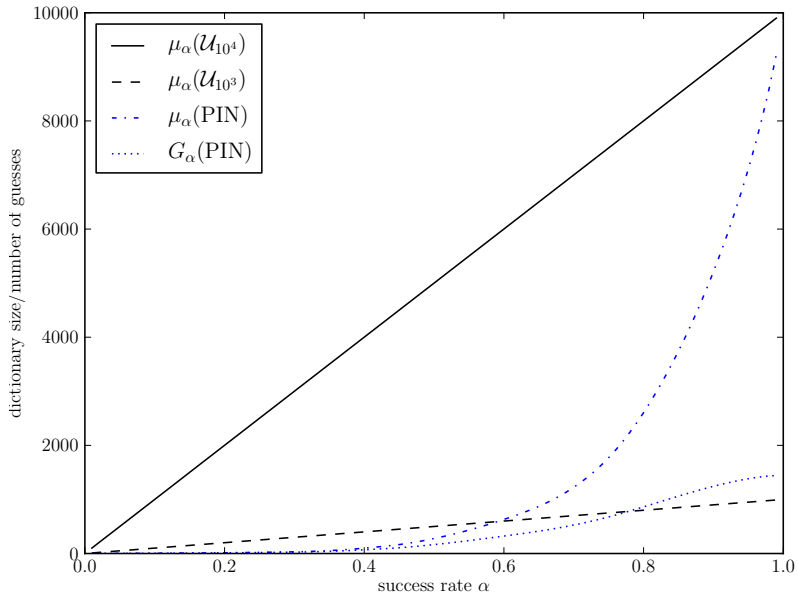
A visual example of the utility of our metrics is shown in Figure 3.2. By plotting over the full interval $0 < \alpha \leq 1$, a large amount of information about the difficulty of guessing attacks is compactly displayed. We call this plot a *guessing curve*. The plot of the non-bit-converted metrics μ_α and G_α in Figure 3.2a is more difficult to interpret as the scale hides information for $\alpha \leq \frac{1}{2}$ and the difference between $\mathcal{U}_{10^4}$ and $\mathcal{U}_{10^3}$ (represented in their slopes) is difficult to evaluate. Plotting the converted metrics $\tilde{\mu}_\alpha$ and $\tilde{G}_\alpha$ (Figure 3.2b) solves both problems, allowing visualisation of the difficulty of guessing a human-chosen PIN over all possible success rates and providing simple comparison with different sizes uniform distributions, whose guessing curves are horizontal lines.

3.2.6 Example calculation

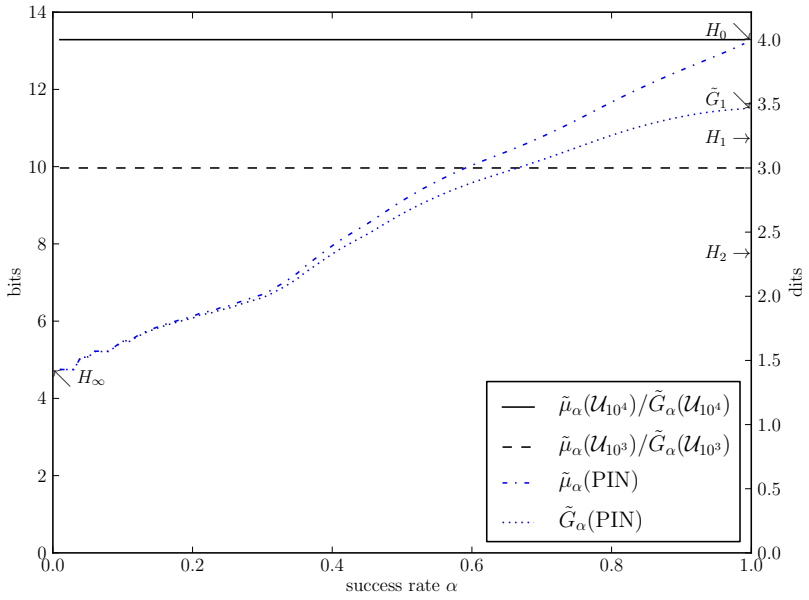
To exercise our metrics, we can compute them for a toy distribution $\mathcal{Z}$ with probabilities $P_\mathcal{Z} = \{\frac{1}{4}, \frac{1}{10}, \frac{1}{10}, \frac{1}{20}, \frac{1}{20} \dots\}$. Regardless of the tail probabilities, an attacker will have a probability $\frac{1}{2}$ of success after making four guesses, giving $\lambda_4(\mathcal{Z}) = \frac{1}{2}$. The uniform distribution $\mathcal{U}_8$ also has $\lambda_4(\mathcal{U}_8) = \frac{1}{2}$, so these two distributions are equivalent with respect to λ_4. Since $\lg|\mathcal{U}_8| = \lg 8 = 3$, we expect $\tilde{\lambda}_4(\mathcal{Z}) = 3$, which we can verify with our formula:

$$\tilde{\lambda}_4(\mathcal{Z}) = \lg\left(\frac{4}{\lambda_4(\mathcal{Z})}\right) = \lg\left(\frac{4}{\frac{1}{2}}\right) = \lg 8 = 3$$

An attacker seeking an minimum success rate $\alpha = \frac{1}{2}$ will similarly need to make four guesses, giving a value of $\mu_{\frac{1}{2}} = 4$. This is equivalent to a uniform distribution $\mathcal{U}_8$ which would have



(a) μ_α, G_α (number of guesses)



(b) μ_α, G_α (effective key-length)

Figure 3.2: Two ways of comparing the guessing difficulty of user-chosen 4-digit PINs from the iPhone data set (§4.1) against uniform distributions of size 10,000 and 1,000 ($\mathcal{U}_{10^4}$ and $\mathcal{U}_{10^3}$, respectively). Fig. 3.2a plots the dictionary size μ_α needed to have a chance of success α as well as the expected number of guesses per account G_α. Fig. 3.2b converts both metrics into an effective key-length, enabling visual comparison across the entire interval $0 \leq \alpha \leq 1$. Traditional single-point metrics H_0, H_1, H_2, H_∞ and $\tilde{G}_1$ are also marked for comparison. Note that $\tilde{\mu}_\alpha$ and $\tilde{G}_\alpha$ are horizontal lines for uniform distributions; an attacker gains no efficiency advantage at a lower desired success rate α.

$\mu_{\frac{1}{2}}(\mathcal{U}_8) = 4$, so we expect that $\tilde{\mu}_{\frac{1}{2}} = \lg|\mathcal{U}_8| = 3$, which we can again verify by our formula:

$$\tilde{\mu}_{\frac{1}{2}}(\mathcal{Z}) = \lg\left(\frac{\mu_{\frac{1}{2}}(\mathcal{Z})}{\lambda_{\mu_{\frac{1}{2}}(\mathcal{Z})}(\mathcal{Z})}\right) = \lg\left(\frac{4}{\lambda_4(\mathcal{Z})}\right) = \lg\left(\frac{4}{\frac{1}{2}}\right) = \lg 8 = 3$$

Note that in this case, four guesses yields a success rate of exactly $\alpha = \frac{1}{2}$. However, if we want a success rate of $\alpha = \frac{1}{2} + \varepsilon$ for any $0 < \varepsilon \leq \frac{1}{20}$, we need 5 guesses. The effect of our smoothed formula is to give the same value for $\tilde{\mu}_\alpha$ for any $\frac{1}{2} < \alpha \leq \frac{1}{2} + \frac{1}{20}$, specifically:

$$\tilde{\mu}_{\frac{1}{2}+\varepsilon}(\mathcal{Z}) = \lg\left(\frac{\mu_{\frac{1}{2}+\varepsilon}(\mathcal{Z})}{\lambda_{\mu_{\frac{1}{2}+\varepsilon}(\mathcal{Z})}(\mathcal{Z})}\right) = \lg\left(\frac{5}{\lambda_5(\mathcal{Z})}\right) = \lg\left(\frac{5}{\frac{11}{20}}\right) = \lg\frac{100}{11} \approx 3.18$$

Computing $G_{\frac{1}{2}}$ is slightly more complicated:

$$\begin{aligned} G_{\frac{1}{2}}(\mathcal{Z}) &= (1 - \lceil\alpha\rceil) \cdot \mu_\alpha(\mathcal{Z}) + \sum_{i=1}^{\mu_\alpha(\mathcal{Z})} p_i \cdot i \\ &= \frac{1}{2} \cdot 4 + \left(\frac{1}{4} \cdot 1 + \frac{1}{10} \cdot 2 + \frac{1}{10} \cdot 3 + \frac{1}{20} \cdot 4\right) \\ &= \frac{59}{20} \\ &= 2.95 \end{aligned}$$

Converting to bits, we get:

$$\begin{aligned} \tilde{G}_{\frac{1}{2}}(\mathcal{Z}) &= \lg\left[\frac{2 \cdot G_\alpha(\mathcal{Z})}{\lceil\alpha\rceil} - 1\right] - \lg(2 - \lceil\alpha\rceil) \\ &= \lg\left[\frac{2 \cdot 2.95}{\frac{1}{2}} - 1\right] - \lg 1.5 \\ &= \lg\left(\tfrac{108}{15}\right) \\ &\approx 2.848 \end{aligned}$$

Note that the result is only slightly lower than $\tilde{\mu}_{\frac{1}{2}}(\mathcal{Z}) = 3$; for this small distribution the ability to stop early doesn't significantly speed up a guessing attack with $\alpha = \frac{1}{2}$.

3.3 Relationship between metrics

3.3.1 Equivalences and simple bounds

We enumerate a few useful relationships between different metrics in Table 3.1. Note that for a discrete uniform distribution $\mathcal{U}_N$, all of the metrics H_n, $\tilde{G}_\alpha$, $\tilde{\lambda}_\beta$ and $\tilde{\mu}_\alpha$ are equivalent. This explains why partial guessing metrics haven't been needed in cryptographic literature, as they provide no additional information for uniform distributions.

equivalences	
$\forall_n \quad H_n(\mathcal{U}_N) = \lg N$	all metrics equal for $\mathcal{U}$
$\forall_\beta \quad \tilde{\lambda}_\beta(\mathcal{U}_N) = \lg N$	all metrics equal for $\mathcal{U}$
$\forall_\alpha \quad \tilde{\mu}_\alpha(\mathcal{U}_N) = \lg N$	all metrics equal for $\mathcal{U}$
$\forall_\alpha \quad \tilde{G}_\alpha(\mathcal{U}_N) = \lg N$	all metrics equal for $\mathcal{U}$
$H_0 = \tilde{\mu}_1 = \tilde{\lambda}_N = \lg N$	metrics depending only on N
$H_\infty = \tilde{\mu}_{\alpha \leq p_1} = \tilde{\lambda}_1 = -\lg p_1$	metrics depending only on p_1
bounds	
$H_\infty \leq \tilde{G}_\alpha, \tilde{\mu}_\alpha, \tilde{\lambda}_\beta$	H_∞ is absolute lower bound
$\tilde{G}_\alpha, \tilde{\mu}_\alpha, \tilde{\lambda}_\beta \leq H_0$	H_0 is absolute upper bound
$\tilde{G}_\alpha \leq \tilde{\mu}_\alpha$	proved in §B.2
$\tilde{G}_\alpha - \tilde{\mu}_\alpha \leq \lg(1 - \lceil \alpha \rceil)$	proved in §B.2
monotonicity	
$H_\infty \leq \ldots \leq H_1 \leq H_0$	H_n decreasing with n
$\tilde{\lambda}_\beta \leq \tilde{\lambda}_{\beta+\varepsilon}$	$\tilde{\lambda}_\beta$ increasing with β
$\tilde{\mu}_\alpha \leq \tilde{\mu}_{\alpha+\varepsilon}$	$\tilde{\mu}_\alpha$ increasing with α
$\tilde{G}_\alpha \leq \tilde{G}_{\alpha+\varepsilon}$	$\tilde{G}_\alpha$ increasing with α

Table 3.1: Relations between guessing metrics

3.3.2 Relationship between H_1 and $\tilde{G}_1$

Several works have developed bounds between H_1 and G_1. Massey was perhaps the first to study the relationship [201], proving a result which is very easy to express in our notation:

$$\tilde{G}_1 \geq H_1 - 1 \tag{3.17}$$

Coupled with the trivial upper bound, Massey's bound gives us $H_0 \geq \tilde{G}_1 \geq H_1 - 1$. A loose bound[9] in the other direction was developed by McEliece and Yu [206]:

$$H_1 \geq \lg N \cdot \frac{2^{\tilde{G}_1} - 1}{N - 1} \tag{3.18}$$

This bound is satisfied with equality when $\tilde{G}_1 = \lg N$ (such as a uniform distribution) because $H_1 \leq \lg N$. For lower values of $\tilde{G}_1$ the bound provides a non-trivial lower bound on H_1.

Massey's bound demonstrates an apparently useful relationship between H_1 and G: it is not possible for a high-entropy distribution to have low guesswork. This has been used to justify H_1 as a guessing metric [168]. As demonstrated in §3.1.3 though, $\tilde{G}_1$ can be skewed beyond utility by outliers. In the next section we'll prove that no useful bound exists between H_1 and $\tilde{G}_\alpha$ for $\alpha < 1$.

[9] This bound was later tightened by de Santis et al. [79] in a much more complicated form.

3.3.3 Incomparability of $\tilde{\mu}_\alpha$ with H_1 or $\tilde{G}_1$

The following theorem demonstrates the fundamental incomparability of Shannon entropy and guesswork to partial guessing metrics. Special cases of these results were demonstrated previously [236, 50] but we can use an enhanced proof technique to prove both results generically. We can also trivially extend to $\tilde{G}_\alpha$ using the bound that $\tilde{G}_\alpha \leq \tilde{\mu}_\alpha$.

Theorem 3.3.1. *Given any $\delta > 0$, $\alpha > 0$, there exists a distribution $\mathcal{X}$ such that $\tilde{G}_\alpha(\mathcal{X}) \leq \tilde{\mu}_\alpha(\mathcal{X}) < H_1(\mathcal{X}) - \delta$ and $\tilde{G}_\alpha(\mathcal{X}) \leq \tilde{\mu}_\alpha(\mathcal{X}) < \tilde{G}_1(\mathcal{X}) - \delta$.*

Proof. The central proof technique is to construct a pathological distribution $\mathcal{X} = \alpha \cdot \mathcal{U}_1 + (1-\alpha) \cdot \mathcal{U}_\gamma$ with one likely event and many very unlikely events, giving:

$$\begin{aligned} H_1(\mathcal{X}) &= \sum_{i=1}^{N} -p_i \lg p_i \\ &\geq \gamma \cdot \frac{1-\alpha}{\gamma} \cdot -\lg \frac{1-\alpha}{\gamma} \\ &\geq (1-\alpha)\lg\gamma - (1-\alpha)\lg(1-\alpha) \end{aligned}$$

Because we have $\tilde{\mu}_\alpha(\mathcal{X}) = 1$ by design, if we set γ such that $H_1(\mathcal{X}) \geq \delta + 2$, then we will also have $\tilde{G}_1(\mathcal{X}) > \delta + 1$, following from Massey's bound on $\tilde{G}_1$ (Equation 3.17), completing the proof. Thus, we must solve for γ:

$$\begin{aligned} \delta + 2 &\leq (1-\alpha)\lg\gamma - (1-\alpha)\lg(1-\alpha) \\ (1-\alpha)\lg\gamma &\leq \delta + 2 + (1-\alpha)\lg(1-\alpha) \\ \lg\gamma &\leq \frac{\delta+2}{1-\alpha} + \lg(1-\alpha) \\ \gamma &\geq 2^{\frac{\delta+2}{1-\alpha} + \lg(1-\alpha)} \\ \gamma &\geq (1-\alpha) \cdot 2^{\frac{\delta+2}{1-\alpha}} \end{aligned}$$

This inequality proves that such a pathological distribution is always possible. □

Note that this construction requires $|\mathcal{X}| \in \Theta(2^\delta)$, the result does not hold if we impose sub-exponential limits on $|\mathcal{X}|$. This is a trivial requirement though because $H_1, \tilde{G}_1 \leq H_0 = \lg N$, therefore having a δ-bit gap inherently requires $|\mathcal{X}| \geq 2^\delta$.

We prove analogous results for $\tilde{\lambda}_\beta$ for any β in §B.3.

3.3.4 Inadequacy of a single partial guessing metric

For highly skewed distributions, as are typical for human-chosen data, there may be several very common items which make secret values easy to guess despite the presence of many

unlikely events which inflate its apparent security if measured by H_1 or $\tilde{G}_1$. We may wish to settle on some single value of α to use with $\tilde{\mu}_\alpha$ or $\tilde{G}_\alpha$ instead. However, we can prove an arbitrary gap is possible between $\tilde{\mu}_{\alpha_1}$ and $\tilde{\mu}_{\alpha_2}$ or $\tilde{G}_{\alpha_1}$ and $\tilde{G}_{\alpha_2}$ for any $\alpha_2 > \alpha_1$:

Theorem 3.3.2. *Given any $\delta > 0, \alpha_1 > 0$, and $\alpha_2 > 0$ with $0 < \alpha_1 < \alpha_2 < 1$, there exists a distribution $\mathcal{X}$ such that $\tilde{\mu}_{\alpha_1}(\mathcal{X}) \leq \tilde{\mu}_{\alpha_1}(\mathcal{X}) - \delta$.*

Proof. This theorem is proved similarly to Theorem 3.3.1 by constructing a pathological mixture distribution:

$$\mathcal{X} = \alpha_1 \cdot \mathcal{U}_1 + (\alpha_2 - \alpha_1) \cdot \mathcal{U}_{2^\delta} + (1 - \alpha_2) \cdot \mathcal{U}_{\frac{1-\alpha_2}{\alpha_2 - \alpha_1} \cdot 2^\delta}$$

By design this gives $\tilde{\mu}_{\alpha_1}(\mathcal{X}) = \lg\left(\frac{1}{\alpha_1}\right) = -\lg \alpha_1$, and furthermore we can show that:

$$\begin{aligned}
\tilde{\mu}_{\alpha_2}(\mathcal{X}) &= \lg\left(\frac{2^\delta + 1}{\alpha_2}\right) \\
&> \lg\left(\frac{2^\delta}{\alpha_1}\right) \\
&= \delta - \lg \alpha_1
\end{aligned}$$

This gives the desired gap δ with $|\mathcal{X}| \in \Theta(2^\delta)$. □

Theorem 3.3.3. *Given any $\delta > 0, \alpha_1 > 0$, and $\alpha_2 > 0$ with $0 < \alpha_1 < \alpha_2 < 1$, there exists a distribution $\mathcal{X}$ such that $\tilde{G}_{\alpha_1}(\mathcal{X}) \leq \tilde{G}_{\alpha_2}(\mathcal{X}) - \delta$.*

Proof. Using our general bounds between $\tilde{G}_\alpha$ and $\tilde{\mu}_\alpha$, we know that $\tilde{G}_{\alpha_1}(\mathcal{X}) \leq \tilde{\mu}_{\alpha_1}(\mathcal{X})$ and $\tilde{G}_{\alpha_2}(\mathcal{X}) \geq \tilde{\mu}_{\alpha_2}(\mathcal{X}) + \lg(1 - \lceil\alpha_2\rceil)$. Therefore, if we set $\delta' = \delta - \lg(1 - \lceil\alpha_2\rceil)$ we can construct a distribution $\mathcal{X}$ such that $\tilde{\mu}_{\alpha_1}(\mathcal{X}) \leq \tilde{\mu}_{\alpha_1}(\mathcal{X}) - \delta'$ by Theorem 3.3.2. By transitivity:

$$\begin{aligned}
\tilde{G}_{\alpha_1}(\mathcal{X}) &\leq \tilde{\mu}_{\alpha_1}(\mathcal{X}) \\
&\leq \tilde{\mu}_{\alpha_2}(\mathcal{X}) - \delta' \\
&\leq \tilde{G}_{\alpha_2}(\mathcal{X}) - \lg(1 - \lceil\alpha_2\rceil) - \delta' \\
&= \tilde{G}_{\alpha_2}(\mathcal{X}) - \lg(1 - \lceil\alpha_2\rceil) - (\delta - \lg(1 - \lceil\alpha_2\rceil)) \\
&= \tilde{G}_{\alpha_2}(\mathcal{X}) - \delta
\end{aligned}$$

The desired gap is now δ with $|\mathcal{X}| \in \Theta\left(2^{\delta - \lg(1 - \lceil\alpha_2\rceil)}\right)$. □

These theorems demonstrate that on top of the inadequacy of H_1 and $\tilde{G}_1$, there is no single measure value of α which is adequate for all security purposes and that α should be chosen to reflect a realistic attacker in a specific environment. Plotting the guessing curve is useful in demonstrating security across all possible α.

3.3.5 Non-additivity of partial guessing metrics

Finally, a major drawback is that none of the partial guessing metrics $\tilde{\lambda}_\beta$, $\tilde{\mu}_\alpha$ or $\tilde{G}_\alpha$ are additive like H_1. It would be convenient if an attacker needing to guess both $X \xleftarrow{\mathrm{R}} \mathcal{X}$ and $Y \xleftarrow{\mathrm{R}} \mathcal{Y}$ at the same time would have $\tilde{\lambda}_\beta(X,Y) \approx \tilde{\lambda}_\beta(\mathcal{X}) + \tilde{\lambda}_\beta(\mathcal{Y})$. Unfortunately, this does not hold:

Theorem 3.3.4. *For any $k \geq 1$, $\beta \geq 1$ and $\varepsilon > 0$, there exists a sequence of distributions $\mathcal{X}_1, \ldots, \mathcal{X}_k$ such that $\tilde{\lambda}_\beta(\mathcal{X}_1, \ldots, \mathcal{X}_k) \leq \max\limits_{1 \leq i \leq k} \left\{ \tilde{\lambda}_\beta(\mathcal{X}_i) \right\} + \varepsilon$.*

Theorem 3.3.5. *For any $k \geq 1$, $0 < \alpha < 1$ and $\varepsilon > 0$, there exists a sequence of distributions $\mathcal{X}_1, \ldots, \mathcal{X}_k$ such that $\tilde{\mu}_\alpha(\mathcal{X}_1, \ldots, \mathcal{X}_k) \leq \max\limits_{1 \leq i \leq k} \{ \tilde{\mu}_\alpha(\mathcal{X}_i) \} + \varepsilon$.*

Theorem 3.3.6. *For any $k \geq 1$, $0 < \alpha < 1$ and $\varepsilon > 0$, there exists a sequence of distributions $\mathcal{X}_1, \ldots, \mathcal{X}_k$ such that $\tilde{G}_\alpha(\mathcal{X}_1, \ldots, \mathcal{X}_k) \leq \max\limits_{1 \leq i \leq k} \left\{ \tilde{G}_\alpha(\mathcal{X}_i) \right\} + \varepsilon$.*

We present proofs for these theorems in §B.4. As a result, guessing two values may be only negligibly more difficult than guessing one even if the values are drawn independently. This problem is worse for correlated variables, which is likely if the same person has chosen both.

3.4 Application in practical security evaluation

For an online attacker we can use $\tilde{\lambda}_\beta$ with β equal to the guessing limits imposed by the system. There is no standard for β, with 10 guesses recommended by usability studies [54], 3 by FIPS guidelines [58], and a variety of values (often ∞) seen in practice [47]. Sophisticated rate-limiting schemes may allow a probabilistic number of guesses [19]. We consider $\tilde{\lambda}_{10}$ a reasonable benchmark for resistance to online guessing, though $\tilde{\lambda}_1 = H_\infty$ is a conservative choice as a lower bound for all metrics proposed.

The separation results of §3.3 mean that for brute-force attacks where an adversary is limited only by time and computing power we can't rely on any single value of α; each value provides information about a fundamentally different attack scenario. For a complete picture, we can compare $\tilde{\mu}_\alpha$ or $\tilde{G}_\alpha$ across all values of α using the guessing curve, as plotted in Figure 3.2.

We might consider $\tilde{\mu}_\alpha$ or $\tilde{G}_\alpha$ for a standard value such as 0.5 as a benchmark ($\tilde{\mu}_{0.5}$ was originally suggested by [50]). While $\tilde{G}_\alpha$ more directly measures the efficiency of a guessing attack, $\tilde{\mu}_\alpha$ can be advantageous in practice because it is simpler to compute. In particular, it can be computed using previously published cracking results reported as "a dictionary of size μ compromised a fraction α of available accounts," as plotted in Figure 2.1b. Furthermore, the difference between the metrics is only significant for higher values of α; for $\alpha \leq 0.5$ the two will never differ by more than 1 bit (from the bound in Table 3.1).

12345*? That's amazing! I have the same combination on my luggage.*

—Mel Brooks as President Skroob, *Spaceballs*, 1987

Chapter 4

Guessing difficulty of PINs

We now turn our attention to 4-digit Personal Identification Numbers, or PINs, which are used to authenticate trillions of pounds in payment card transactions annually. They are also used in a variety of other security applications where the lack of a full keyboard prevents the use of text passwords such as electronic door lock codes, smartphone unlock codes and voicemail access codes. Despite their importance, to date no research has examined the difficulty of guessing human-chosen PINs.[1]

PINs are an easy first application for our guessing metrics because the distribution is limited to the set $\{\texttt{0000}, \ldots, \texttt{9999}\}$, with the total number of events limited to $N = 10,000$. This makes sample size less problematic than it is for distributions like passwords which are drawn from a theoretically infinite domain. However, we don't have direct data on the distribution of human-chosen banking PINs. Instead, we will study two other distributions of 4-digit sequences and use a large survey to estimate the distribution of banking PINs.

Because it is computationally trivial to search the entire space PINs, modeling offline attacks using metrics like $\tilde{G}_{0.5}$ is mainly of theoretical interest. We are more interested in λ_3 and λ_6, given that attackers can typically try 3 PIN guesses at an ATM machine and 3 at a CAP reader (§2.1.2).

4.1 Human choice of other 4-digit sequences

We begin with two non-banking sources of human-chosen, secret 4-digit sequences:

Sequences occurring in text passwords

Many users include a consecutive sequence of exactly 4 digits within a text password. We extracted all such sequences from the RockYou password distribution and call the resulting

[1] Bias in the distribution of bank-chosen PINs has been show due to skewed decimalisation tables [178, 41].

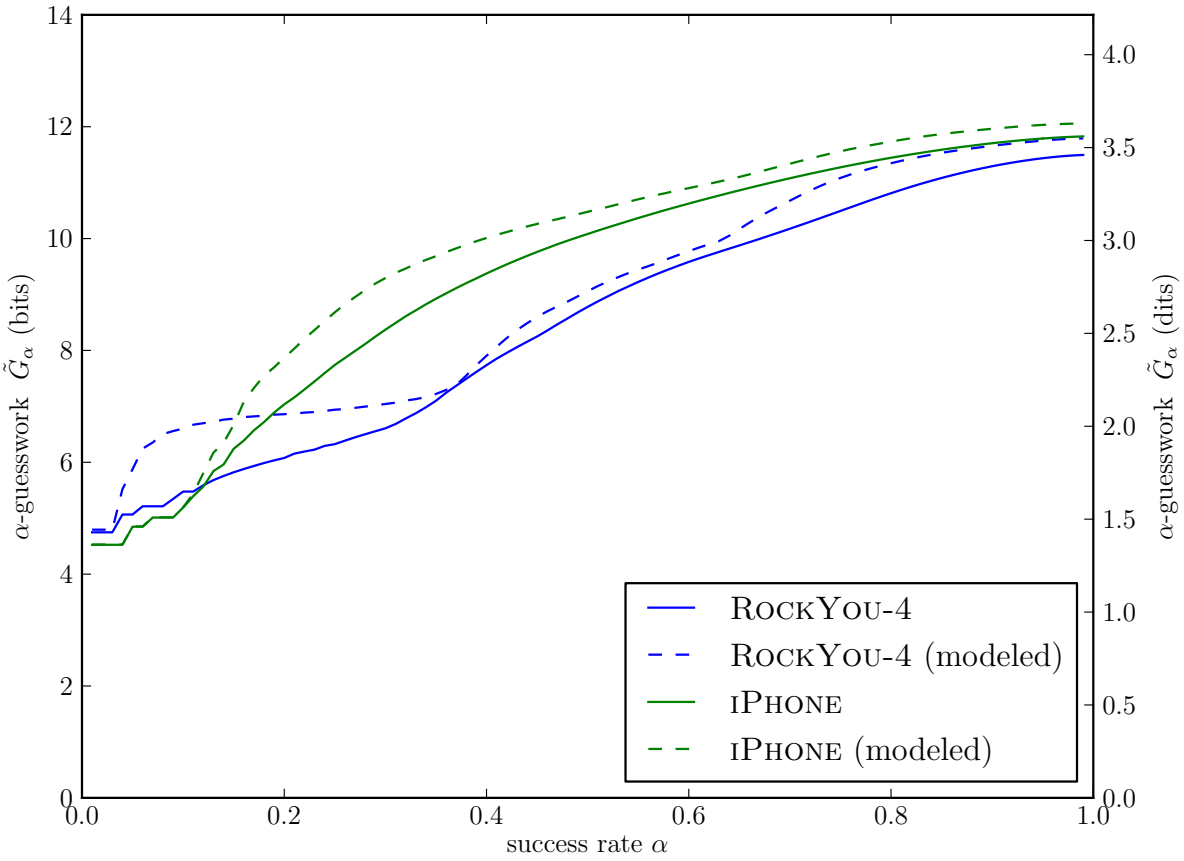


Figure 4.1: Guessing curves for the 4-digit sequences found in RockYou passwords and iPhone unlock codes, as well as the best-fit model distributions produced by linear regression on 25 variables (see Table 4.2).

distribution RockYou-4. The 1,778,095 observed sequences include every 4-digit sequence, from `1234` with 66,193 occurrences (3.7%) to `8439` with just 10 occurrences (0.0006%).

Though these sequences occurred as part of longer strings, a manual inspection of 100 random passwords containing a 4-digit sequence identified only 3 with an obvious connection between the digits and text (`feb1687`, `classof2007` and `2003chevy`), suggesting that digits and letters are often semantically independent. Users also show a particular affinity for 4-digit sequences, using them significantly more often than 3-digit sequences (1,599,959) or 5-digit sequences (497,791).

Smartphone unlock codes

Our second data set was published (in aggregate form) in June 2011 by Daniel Amitay, an iPhone application developer who deployed a screen locking mechanism which requires a 4-digit sequence to unlock. This data set, which we'll call iPhone, was much smaller with only 204,508 PINs. It doesn't support reliable estimates of low-frequency PINs, as 46 possible PINs

distribution	H_1	$\tilde{G}_1$	$\tilde{G}_{0.5}$	λ_3	λ_6
RockYou-4	10.74	11.50	8.78	8.04%	12.29%
RockYou-4 (modeled)	11.01	11.79	9.06	5.06%	7.24%
iPhone	11.42	11.83	10.08	9.23%	12.39%
iPhone (modeled)	11.70	12.06	10.48	9.21%	11.74%
$\mathcal{U}_{10^4}$ (random PIN)	13.29	13.29	13.29	0.03%	0.06%

Table 4.1: Guessing metrics for the 4-digit sequences in RockYou passwords and iPhone unlock codes. Values are also shown for the regression-model approximation for each distribution and for a uniform distribution of 4-digit PINs.

weren't observed at all. `1234` was again the most common, representing 4.3% of all PINs. The screen unlock codes were entered using a square number pad very similar to standard PIN-entry pads. Geometric patterns, such as PINs consisting of digits which are adjacent on the keypad, were far more common than in the RockYou-4 set.

The guessing curves for the two distributions are plotted in Figure 4.1; other metrics are listed in Table 4.1. The security of both distributions drops below 2 dits against attackers that require only a 20% success rate, meaning that in some situations these PINs are easier to guess than random 2-digit numbers.

4.1.1 Estimating factors influencing user choice

Plotting the RockYou-4 distribution in a 2-dimensional grid (Figure 4.2) highlights some basic factors influencing user choice. The most prominent features are the stripe of recent years and the set of calendar dates in `MMDD` and `DDMM` format, which trace the variation in lengths of each month. Many other features, such as a diagonal line of PINs with the same first and last two digits and a horizontal line of PINs ending in `69`, can be clearly seen.

To quantitatively study factors affecting PIN selection, we model the PIN distribution as a mixture distribution of simpler distributions arising from human-level PIN selection stratgies. For example, the strategy "repeat the same digit four times" assigns probability $\frac{1}{10}$ to each of {`0000`, ..., `9999`} and probability 0 to all other PINs. Each observed PIN frequency is:

$$f_i = \varepsilon_i + \sum_{1 \le j \le N} p_i^j \cdot w_j \tag{4.1}$$

where p_i^j is the observed popularity of PIN i in distribution j (expressed as a proportion of all PINs observed), w_j is the weight of distribution j in the mixture, and ε_i is a residual error term for each observed PIN. Given this model and mathematical definitions of common pin-selection strategies, we can estimate the weights w_j using multiple regression [195], solving for the weights which minimizes the sum of squared residuals $\sum \varepsilon_i^2$.

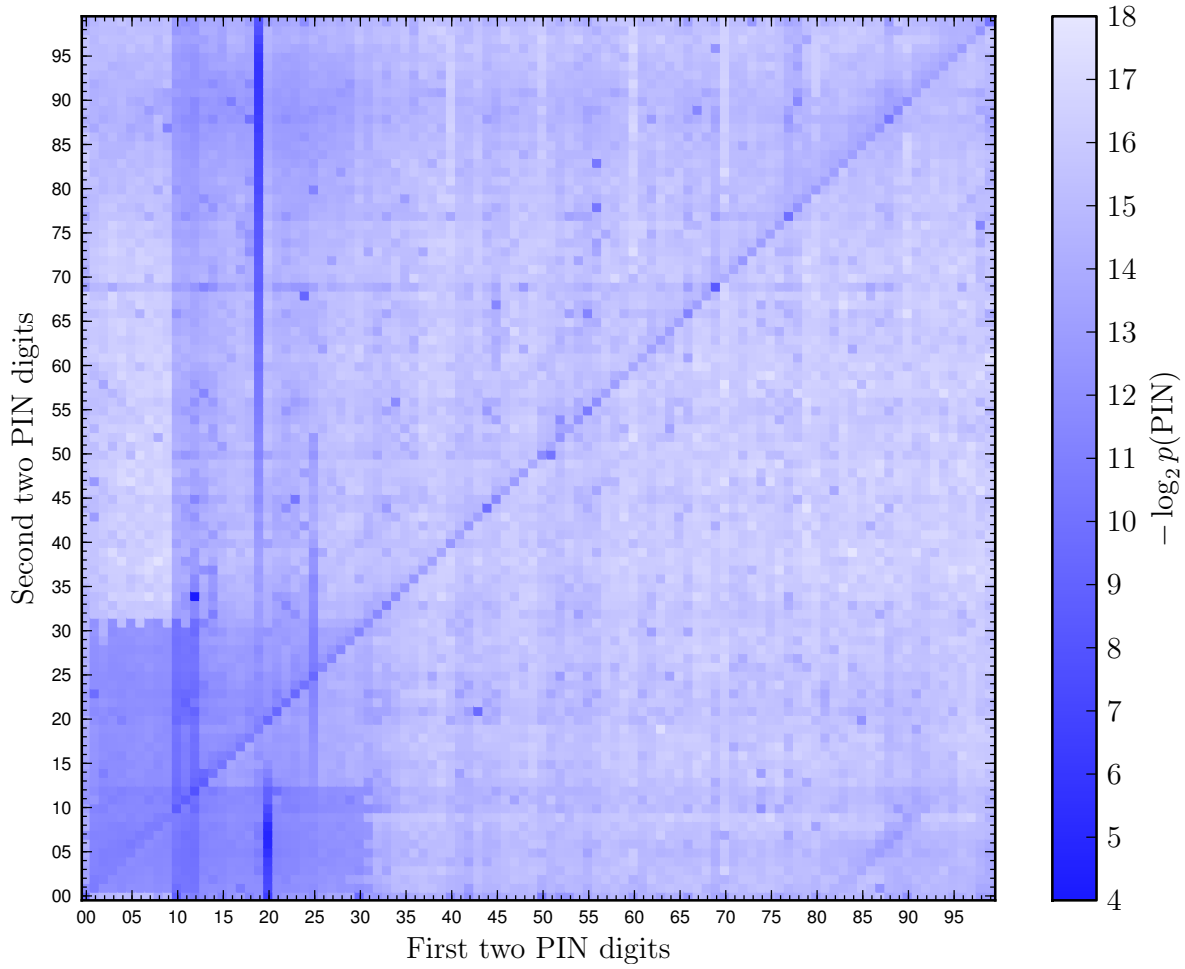


Figure 4.2: The distribution of 4-digit sequences within RockYou passwords (RockYou-4). Each cell shows the frequency of an individual PIN.

Assuming our PIN selection strategies are mutually exclusive, the weights can be interpreted as the proportion of the population employing each strategy. Our process for identifying relevant input functions was iterative: we began with a single strategy representing random selection in which each PIN is equally likely and progressively added functions which could explain the PINs which were the most poorly fit. We measured the fit of our model using the adjusted coefficient of determination $\bar{R}^2$ [293], which corrects for the number of input variables in the model so that adding random input variables will not increase the indicated fit of the model. We stopped when we could no longer identify intuitive functions which increased $\bar{R}^2$ [293]. We further tested our final set of input functions for redundancy by removing each one and confirming that $\bar{R}^2$ decreased.

We were cautious to avoid over-fitting the training data sets, particularly for PINs which represent recent years, shown in Figure 4.3. The popularity of recent years has peaks between the current year and the year 1990. This interval probably represents recent events like grad-

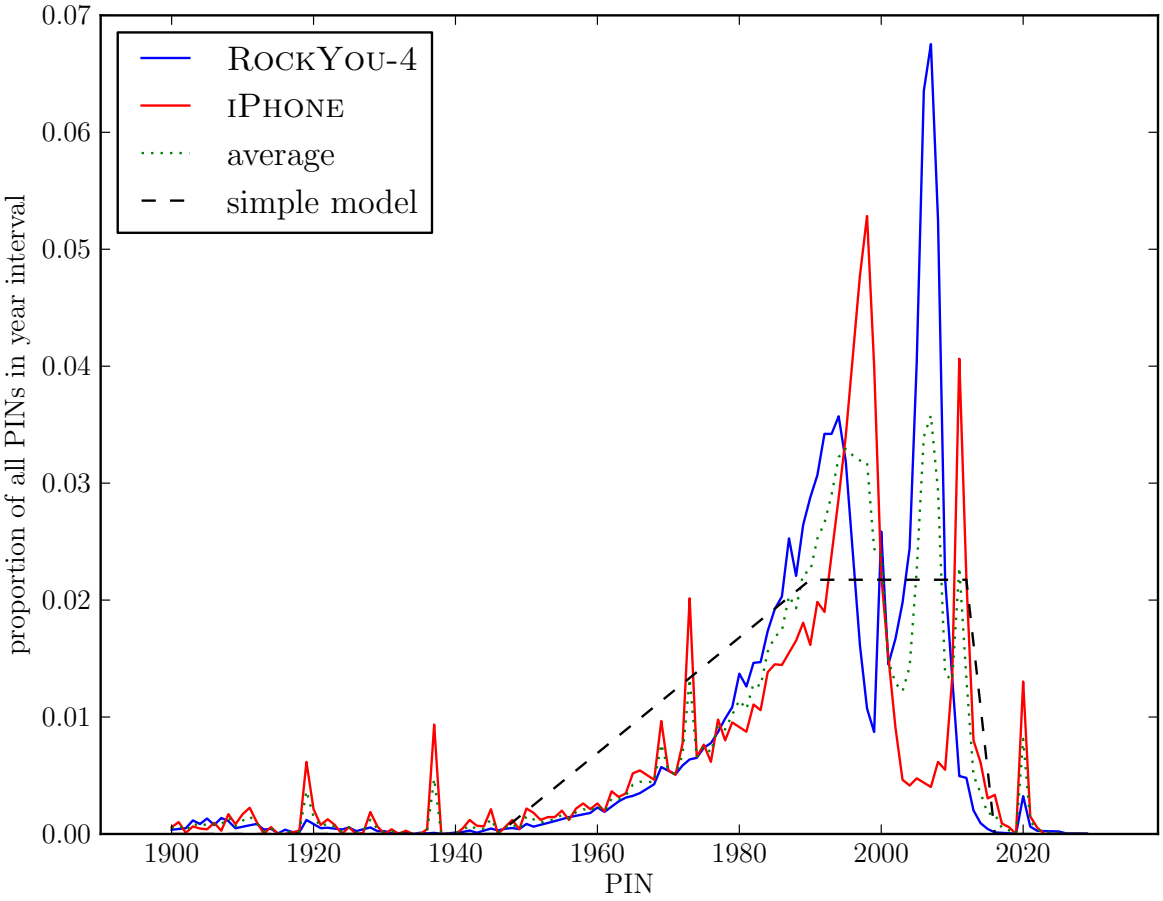


Figure 4.3: Frequency of PINs in the interval from 1900–2025 and a simplified model to reduce over-fitting. Some outliers demonstrate known confounding factors: `1937` and `1973` represent the four-corners of a numeric keypad, `1919` and `2020` are repeated digit pairs.

uations or marriages (or perhaps registration). There is steady decline for older years, likely due to the drop-off in frequency of birthdays and events which are still memorable. Due to the large fluctuations for recent years in both data sets and a possibly younger demographic for both data sets compared to the general population, we used an intentionally biased model for the popularity of different years in PIN selection: constant popularity for all years in the past 20 years and linear drop-offs for years from 20–65 years in the past and for 5 years into the future. This model, plotted in Figure 4.3, was used for PINs representing 4-digit years directly as well as `DMYY` and `MMYY` PINs.

After fixing the year model, we removed the interval of years from the regression model to avoid skewing the model's estimation of other parameters to correct for the intentionally weakened model of the year distribution. We similarly added singleton input functions for `1234`, `0000`, `1111`, and `2580` to avoid omitted-variable bias caused by these significant outliers.

The complete results of our final model with 25 input functions are shown in Table 4.2. All of the input functions were binary, except for years, calendar dates (in which Feb. 29$^{\text{th}}$ was

factor	example	RockYou-4	iPhone	surveyed
		date		
DDMM	2311	5.26	1.38	3.07
DMYY	3876	9.26	6.46	5.54
MMDD	1123	10.00	9.35	3.66
MMYY	0683	0.67	0.20	0.94
YYYY	1984	33.39	7.12	4.95
total		58.57	24.51	22.76
		keypad		
adjacent	6351	1.52	4.99	—
box	1425	0.01	0.58	—
corners	9713	0.19	1.06	—
cross	8246	0.17	0.88	—
diagonal swipe	1590	0.10	1.36	—
horizontal swipe	5987	0.34	1.42	—
vertical swipe	8520	0.06	4.28	—
spelled word	5683	0.70	8.39	—
total		3.09	22.97	8.96
		numeric		
ending in 69	6869	0.35	0.57	—
digits 0-3 only	2000	3.49	2.72	—
digits 0-6 only	5155	4.66	5.96	—
repeated pair	2525	2.31	4.11	—
repeated quad	6666	0.40	6.67	—
sequential down	3210	0.13	0.29	—
sequential up	4567	3.83	4.52	—
total		15.16	24.85	4.60
random selection	3271	23.17	27.67	63.68

Table 4.2: Results of linear regression. The percentage of the variance explained by each input function is shown for the RockYou-4 and iPhone data sets. The final column shows estimates for the prevalence of each category from our user survey.

discounted), and words spelled on a keypad.[2] All of the input functions we chose contributed positively to the probability of a PIN being selected, making it plausible to interpret the weight assigned to each input function as the proportion of the population choosing a PIN by

[2]We used the distribution of four-letter passwords in the RockYou data set to approximate words used in spelled-out PINs. `love` was the most common 4-letter password by a large margin and its corresponding PIN `5683` was a significant outlier.

each method. The intercept term fits this interpretation naturally as the proportion of users choosing a random PIN. This simple model was able to fit both distributions quite accurately: the coefficient of determination $\bar{R}^2$ was 0.79 for RockYou-4 and 0.93 for iPhone. Under the conventional interpretation, this means the model explained 79% and 93% of the variance.

Additional confirmation of our model comes from its accurate approximation of guessing metrics in the underlying distributions, as seen in Figure 4.1 and Table 4.1. The model consistently provides an over-approximation by about 0.2–0.3 bits (< 0.1 dit) indicating that the inaccuracy is mainly due to missing some additional biases in user selection. This is acceptable for estimating an upper bound on the guessing difficulty of banking PINs.

4.2 Surveying banking PIN choices

Lacking empirical data on real banking PINs, we use a survey to assess user behaviour. The low frequency of many PINs means a survey of hundreds of thousands of users would be needed to observe all PINs and some users might feel uncomfortable disclosing their PINs. We addressed both problems by asking users only if their PINs fall into the generic categories identified by our regression model.

We deployed our survey online using the Amazon Mechanical Turk platform, a crowd-sourcing marketplace for short tasks. The survey was advertised as "Short research survey about banking security" intended to take five minutes. We deliberately displayed the University of Cambridge as the responsible body to create a trust effect.

About half of Mechanical Turk "workers" are US residents with a valid bank account registered for payment and tax purposes; the remainder (predominantly residents of India) can only be paid in Amazon gift cards [150]. We limited participation to only US residents with bank details registered in order to make repeat participation difficult.[3] The survey was piloted on 20 respondents. We removed demographic questions[4] after pilot respondents indicated this made them feel less comfortable discussing their PINs. We then administered the final survey to 1,351 respondents and kept 1,337 responses.[5] Respondents were paid US$0.10–0.44 including bonuses for complete submission and thoughtful feedback.

4.2.1 PIN usage characteristics

A total of 1,177 respondents reported using numeric banking PINs and were asked a series of questions about their PIN usage. A summary of the question phrasing and responses is

[3] Repeat participation is still possible through fraudulently-obtained banking accounts or if a valid account is created and the credentials given to another person. We relied on Amazon's policing of such fraud.

[4] Demographics of American Mechanical Turk workers are close to those of the general Internet-using public, though slightly younger, more female, and lower-income [150].

[5] It is common practice on Mechanical Turk to include careful test questions to eliminate respondents who have not diligently read the instructions. Our test question involved using PINs to unlock a unicorn shed (§C).

provided in §C. A surprising number (about 19%) of users rarely or never use their PIN, relying on cash or cheques and in-person interaction with bank tellers. Several participants reported in feedback that they distrust ATM security to the point that they don't even know their own PINs. Many others stated that they prefer signature verification to typing in their PIN. However, 41% of participants indicated that PINs were their primary authentication method for in-store payments, with another 16% using PINs or signatures equally often. Of these users, nearly all (93%) used their PINs on at least a weekly basis.

Over half of users (53%) reported sharing their PIN with another person, though this was almost exclusively a spouse, partner, or family member. This is consistent with a 2007 study which found that about half of online banking users share their passwords with a family member [276]. Of the 40% of users with more than one payment card, over a third (34%) reported using the same PIN for all cards. This rate is lower than that for online passwords, where the average password is reused across six different sites [103]. The rate of forgotten PINs was high, at 16%, although this is again broadly consistent with estimates for online passwords, where about 5% of users forget their passwords every 3 months at large websites [103]. Finally, over a third of users (34%) re-purpose their banking PIN in another authentication system. Of these, the most common were voice-mail codes (21%) and Internet passwords (15%).

4.2.2 PIN selection strategies

We invited the 1,108 respondents with a PIN of exactly 4 digits to identify their PIN selection method. This was the most sensitive part of the survey and users were able to not provide this information without penalty, removing a further 27% of respondents and leaving us with 603 responses from which to estimate PIN strength. We presented users with detailed descriptions and examples for each of the selection strategies identified in our regression model. Users were also able to provide free-form feedback on how they chose their PIN. The aggregated results of our survey are shown alongside our regression model in Table 4.2.

The largest difference between our survey results and the regression models was a huge increase in the number of random and pseudo-random PINs: almost 64% of respondents in our survey, compared to 23% and 27% estimated for our example data sets. Of these users, 63% reported that they either used the PIN initially assigned by their bank or a PIN assigned by a previous bank.[6] Another 21% reported the use of random digits from another number assigned to them, usually either a phone number or an ID number from the government, an employer, or a university (about 30% for each source).[7]

[6] We explored the possibility that some participants kept their initial PIN simply because they rarely or never used their card, but the rate was statistically indistinguishable by Fisher's exact test for users using their PIN at least once per week.

[7] While reusing identification numbers and phone numbers in PINs may open a user to targeted attacks, they should appear random to an untargeted guessing attacker.

guessing scenario	H_1	$\tilde{G}_1$	$\tilde{G}_{0.5}$	λ_3	λ_6
BankingPIN	12.90	12.83	12.38	1.44%	1.94%
BankingPIN-BL	13.13	12.95	12.67	0.12%	0.24%
BankingPIN-DOB	12.57	12.80	12.28	5.53%	8.25%
BankingPIN-DOB-BL	12.85	12.92	12.60	5.13%	5.65%
$\mathcal{U}_{10^4}$ (random PIN)	13.29	13.29	13.29	0.03%	0.06%

Table 4.3: Guessing metrics for our estimated banking PIN distribution using the model computed from our survey. The BL and DOB variants model the use of a blacklist by banks and knowledge of the user's birth date by the attacker.

Of users with non-random PINs, dates were by far the largest category, representing about 23% of users, comparable to the iPhone distribution and about half the rate of the RockYou-4 distribution. The choice of date formats was similar to the other data sets with the exception of 4-digit years, which were less common in our survey. We also asked users about the significance of the dates in their PINs: 29% used their own birth date, 26% the birth date of a partner or family member and 25% an important life event like an anniversary or graduation.

Finally, about 9% of users chose a pattern on the keypad and 5% a numeric pattern such as repeated or sequential digits. Our sample size was insufficient to provide an accurate breakdown of users within these categories.

4.3 Approximating banking PIN strength

Using our survey data and regression model we produced an estimated distribution BankingPIN.[8] We list key statistics in Table 4.3. Interestingly, security against offline attacks as measured by any of the metrics $\tilde{G}_{0.5}$, $\tilde{G}_1$, or H_1 is between 12.4 and 12.9 bits (3.7–3.9 dits), close to the maximum possible 13.3 bits/4 dits.

4.3.1 Partial guessing

Banking PINs appear considerably more vulnerable against partial guessing attacks, as seen in the guessing curve in Figure 4.4. As noted in Table 4.3, an attacker with 3 guesses will have a $\lambda_3 = 1.4\%$ chance of success and an attacker with 6 guesses a $\lambda_6 = 1.9\%$ chance of success, equivalent to $\tilde{\lambda}_6 = 8.3$ bits of security (2.5 dits). This is significantly better than the estimates based on the RockYou-4 or iPhone distributions (Table 4.1), for which $\lambda_6 > 10\%$. The optimal guessing order is `1234` followed by `1990`–`1986`.

[8] Within the categories of numeric patterns and keypad patterns, we used the sub-distribution from the iPhone data set due to lack of sufficient sample size.

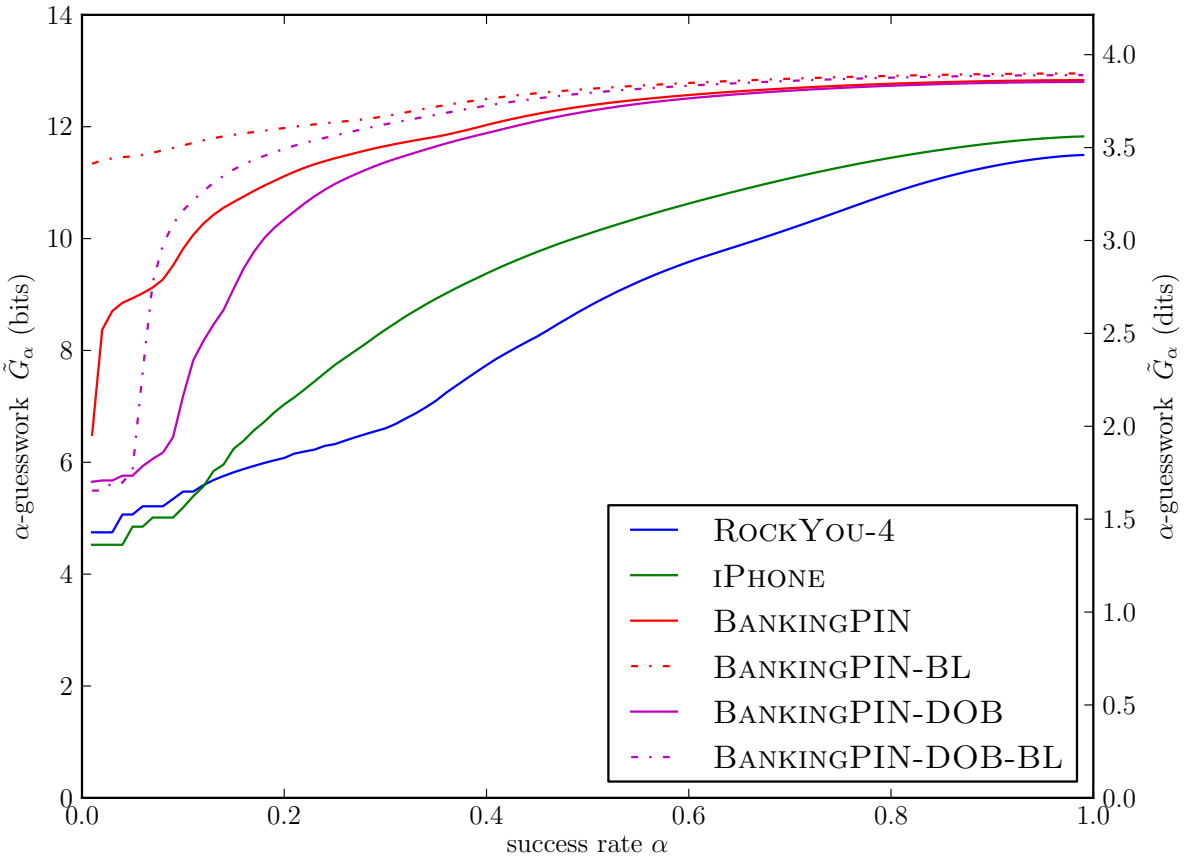


Figure 4.4: Estimated guessing curves for banking PINs based on our survey data and regression model (see Table 4.2). We also model the effective distribution if the adversary knows the user's birth date and if a blacklist of weak PINs is utilised.

4.3.2 Known birth date guessing

Given the large number of users basing their PIN on their birth date (nearly 7%), we evaluated the success of an attacker who can leverage a known birth date, for example if a card is stolen in a wallet along with an identification card. The effects vary slightly with the actual birth date: if variants of the date also correspond to common PINs such as `1212`, the attacker's success rate will be higher. We calculated guessing probabilities for all dates from 1960–1990 and list guessing metrics for the median-difficulty birth date of June 3, 1983 as BankingPIN-DOB. In this scenario, the attacker's optimal strategy shifts to guessing, in order, `1983`, `6383`, `0306`, `0603`, `1234`, and `0683`. As seen in Table 4.3, the attacker improves considerably in this scenario: λ_6 increases to 8.2%, providing only $\tilde{\lambda}_6 = 6.2$ bits (1.9 dits) of security.

4.3.3 Effectiveness of blacklisting

We can model the effects of a modest blacklist of 100 weak PINs. We assume that users attempting to use a blacklisted PIN will be re-distributed randomly according to the prob-

	number of stolen cards				
guessing scenario	1	2	3	4	E_{wallet}
BankingPIN	1.4%	1.9%	2.4%	2.9%	1.7%
BankingPIN-BL	0.1%	0.2%	0.4%	0.5%	0.2%
BankingPIN-DOB	5.5%	8.2%	9.3%	9.7%	6.7%
BankingPIN-DOB-BL	5.1%	5.6%	5.9%	6.0%	5.4%
$\mathcal{U}_{10^4}$ (random PIN)	0.0%	0.1%	0.1%	0.1%	0.0%

Table 4.4: Probability of a successful attack given multiple cards from one person. The final column is an expected value given the observed rate of card ownership.

ability of non-blacklisted PINs being originally chosen, producing a modified distribution BankingPIN-BL. The optimal blacklist is:

`0000`, `0101`–`0103`, `0110`, `0111`, `0123`, `0202`, `0303`, `0404`, `0505`, `0606`, `0707`, `0808`, `0909`, `1010`, `1101`–`1103`, `1110`–`1112`, `1123`, `1201`–`1203`, `1210`–`1212`, `1234`, `1956`–`2015`, `2222`, `2229`, `2580`, `3333`, `4444`, `5252`, `5683`, `6666`, `7465`, `7667`.

The effects are substantial—λ_6 drops to 0.2%, equivalent to $\tilde{\lambda}_6 = 11.6$ bits (3.9 dits) of security, indicating that a very small blacklist nearly eliminates insecurity due to human choice. Unfortunately, as seen in Table 4.3 and Table 4.4, blacklisting is much less effective against known birth date attacks, only reducing λ_6 to 5.1% ($\tilde{\lambda}_6 = 6.9$ bits/2.1 dits). With a small blacklist, it is only possible to block the `YYYY` format, leaving an attacker to try `DDMM`, `MMDD`, etc. Preventing this would require user-specific blacklists.

4.4 Security implications

We are most concerned with the scenario of a thief guessing PINs after stealing a wallet, which may contain multiple cards. We calculate the guessing probability of a thief with multiple stolen cards, for example from an entire wallet or purse, in Table 4.4. Though most of our surveyed users own only one card with a PIN, on expectation stealing a wallet instead of a single card raises a thief's guessing chances by over a third. Because our survey results suggest that virtually all payment card users (99%) carry documentation of their birth date alongside their card,[9] we estimate that a competent thief will gain use of a payment card once every 11–18 stolen wallets, depending on the use of a PIN blacklist.

For randomly assigned PINs ($\mathcal{U}_{10^4}$), this rate would be less than 1 in 1,200 wallets. We thus conclude that in the case of PINs, user selection makes a significant impact on security.

[9] The prevalence of carrying ID varies by locale. In 24 US states carrying ID is legally required. In the UK, carrying ID is not required and fewer citizens carry it.

Young scamels from the rock. Wilt thou go with me?

—Caliban in *The Tempest*, William Shakespeare circa 1611

Chapter 5

Estimation using sampled data

In our initial treatment of guessing difficulty in §3, we assumed complete information is available about the distribution of interest $\mathcal{X}$. In practice, we will typically need to approximate $\mathcal{X}$ using empirical data. In this chapter we explore the challenge of estimating guessing metrics using a finite set of M independent random samples $X_1 \xleftarrow{\text{R}} \mathcal{X}, \ldots, X_M \xleftarrow{\text{R}} \mathcal{X}$. For a given sample, we'll write $V(M)$ for the number of distinct events observed[1] and $V(m, M)$ for the total number of events observed m times. It turns out that for distributions with a theoretically infinite space of events, such as passwords which may be arbitrarily long strings, even with tens of millions of samples we will not be able to compute some properties of the distribution at all and must take care in computing others.

5.1 Naive estimation

The simplest approach is to compute metrics directly on the distribution of samples, which we denote $\hat{\mathcal{X}}$.[2] We call this approach, which we used to evaluate PINs in §4 without explanation, *naive estimation*[3]. Unfortunately this approach produces substantial, systematic underestimates for many guessing metrics of interest.

The easiest way to see this is to take a large data set and compare naive estimates for guessing metrics computed using all available samples with those computed using a random subset, a statistical technique known as *cross-validation*. For some calculations, such as the average length of a password in a data set, cross-validation on a random subsample will produce the correct value on expectation over all possible random samples. Of course, the estimate will have increasing variance in any single random sample as the sample size decreases.

[1] In statistical linguistics, this is called the *vocabulary size* for the sample.

[2] We use the hat symbol ˆ for any metric estimated from sampled data.

[3] Note that this approach is not maximum-likelihood estimation. As we will see §5.4, better maximum-likelihood techniques exist for some properties like individual event probabilities.

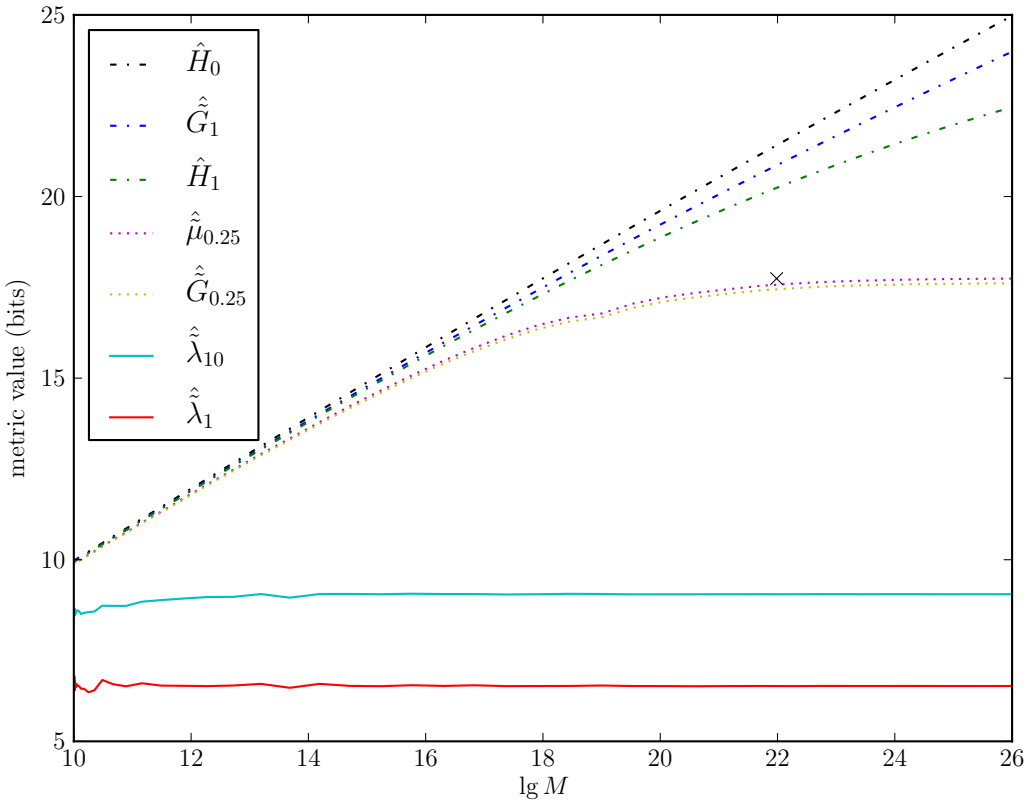


Figure 5.1: Naive estimates of guessing metrics for the YAHOO data set as sample size M increases. Estimates for $\tilde{\lambda}_1$ and $\tilde{\lambda}_{10}$ converge quickly; $\tilde{\mu}_{0.25}$ converges around $M = 2^{22}$ (marked $\times$) as predicted in §5.5. H_0, H_1, and $\tilde{G}_1$ do not converge.

For most guessing metrics, however, cross-validation shows that naive estimates are biased: on expectation naive estimates are lower for a random subsample than the complete data set and the problem is worse for lower M. To demonstrate this visually, we can take a series of subsamples from a large data set and plot our naive estimates for guessing metrics as the size of the subsamples increases. For most of this chapter, we will use the YAHOO data set of 69 M passwords. We will discuss the provenance of this data set further in §6, for now we use it as a canonical large data set of user-chosen passwords.

In Figure 5.1 we plot naive estimates for random subsamples of the YAHOO set. Estimates for $\hat{H}_0$, $\hat{H}_1$, and $\hat{G}_1$ increase continuously with increasing sample size. This demonstrates that naive estimation is inherently biased for metrics which rely on the complete distribution. The essential problem is that many rare events will not be observed in a random subsample. Since the sum of estimated probabilities remains 1, the cumulative probability of all observed events must be an overestimate.

5.2 Known negative results

5.2.1 Estimating $H_0 = \lg N$

Estimating H_0, the upper bound for all guessing metrics, is equivalent to estimating the total number of possible events N. This problem was famously studied by Ronald Fisher, often called the founder of modern statistics, to estimate the total number of species of butterfly in the Malaysian rain forest given a sample of captured butterflies of various species in 1943 [101]. Fisher produced a heuristic formula which implicitly assumed a Zipfian distribution of species abundance. To date, there are no known techniques for estimating N which are both non-parametric (don't assume anything about the underlying distribution) and non-biased.

The problem was also famously studied by Alan Turing, considered the founder of computer science, and his colleague Irving Good during their wartime cryptanalytic work at Bletchley Park. Published after the war 1953 [125], the so-called Good-Turing method (which we will return to in §5.4) can produce non-biased estimates for the cumulative probability of all events not observed in a sample, the so-called "missing mass." This problem can also be cast as the growth rate $\frac{dV(M)}{dM}$ of new events as more samples are taken.

Interestingly, the maximum-likelihood estimation of $\frac{dV(M)}{dM}$ is simply $\frac{V(1,M)}{M}$, the proportion of events in the sample observed only once. An informal proof of this result is that, of all possible orders in which the random sample could have been taken, exactly $\frac{V(1,M)}{M}$ of them would have the last observation be a new event, meaning this is the marginal rate at which new events are still likely to be seen. For the complete YAHOO data set, we have $\frac{V(1,M)}{M} = 42.5\%$, indicating that even at very large M the observed number of events $\hat{N}$ is still growing rapidly.

Regardless of these results, as discussed in §3.1.3 we have $N \geq 2^{128}$ as a lower bound for real passwords given the existence of some machine-chosen pseudorandom strings. Thus, estimating N appears to have no practical security consequence.

5.2.2 Estimating H_1

Estimating the Shannon entropy H_1 from a sample has also been extensively studied. Beirlant et al. provide a good survey of estimation approaches [30]. In 2011 Valiant and Valiant proved an optimal algorithm for determining H_1 to within an additive constant requiring $O\left(\frac{N}{\ln N}\right)$ samples [298]. Given the unbounded value for N for text passwords, this near-linear requirement of samples means estimating H_1 in a non-parametric manner is impossible.

Earlier, Valiant proved [299] that to simply distinguish between distributions $\mathcal{X}_a$ and $\mathcal{X}_b$ with $H_1(\mathcal{X}_a) = a$ and $H_1(\mathcal{X}_b) = b > a$ with probability non-negligibly greater than $\frac{1}{2}$ requires $o\left(N^{\frac{2a}{3b}}\right)$ samples. Even determining if $H_1(\mathcal{X})$ is less than a bits or more than $b = 2a$ bits would require more than $\sqrt[3]{N}$ samples, which would be intractable for real password distributions.

5.2.3 Generic limits on estimating symmetric properties

The proof techniques developed by Valiant to derive the bounds on estimating H_1 above are generic and can be extended to any property of a distribution which is *symmetric* (constant after a re-labeling of all events in the distribution) as are all guessing metrics in our model. Valiant's Low-Frequency Blindness Theorem [299] states that for any symmetric property P, if there exist distributions $\mathcal{X}_a$ and $\mathcal{X}_b$ such that $P(\mathcal{X}_a) = a$ and $P(\mathcal{X}_b) = b > a$ but the probabilities of $\mathcal{X}$ and $\mathcal{Y}$ are only different for events with probability $\leq \frac{1}{n}$, then no estimation algorithm can decide if $P(\mathcal{X}) \leq a$ or $P(\mathcal{X}) \geq b$ with fewer than $O(n)$ samples.

Essentially, this means that events with an expected frequency of observation $f \lesssim 1$ in a sample are useless to any approximation algorithm. The difficulty of reasoning about events observed only once in a sample is well-known in linguistics, where such events are called *hapax legomena* (singular *legomenon*),[4] Greek for "said only once."

Thus, there are fundamental reasons why H_0, H_1 and G_1, which depend on every event in the distribution, cannot be estimated with fewer than $M = O(N)$ samples, which is required to ensure all events are expected to be observed. Fortunately, our partial guessing metrics explicitly don't depend on low-frequency events, meaning that they can be estimated with a much smaller sample using the very simple approach proved asymptotically optimal in Valiant's Canonical Testing Theorem [299]: ignore all low-frequency events, use the naive probability estimates for all more-frequent events, and compute the property of interest directly.

5.3 Sampling error for frequent events

We now focus on partial guessing metrics which can be reliably estimated, beginning with $\tilde{\lambda}_1$ which depends only on p_1, the probability of the most common event. If an event x_i is observed f_i times in a sample, the naive estimate $\hat{p}_i = \frac{f_i}{M}$ will converge to the correct p_i as $M \to \infty$ due to the law of large numbers. If $\sqrt{1 - p_i} \approx 1$, a reasonable assumption for all items in most distributions subject to guessing attacks, the standard error of $\hat{p}_i$ is approximately:

$$\text{stderr}(\hat{p}_i) = \sqrt{\frac{p_i(1-p_i)}{M}} \cdot \frac{1}{p_i} \approx \sqrt{\frac{f_i}{M^2}} \cdot \frac{M}{f_i} = \frac{1}{\sqrt{f_i}} \tag{5.1}$$

We can alternately derive this by approximating our sample as a Poisson process with an expected rate $\lambda = f_i$ of observations of x_i per M samples. Because the standard deviation of a Poisson random variable is $\sqrt{\lambda}$, this gives the same expected error in $\hat{p}_i$:

$$\text{stderr}\left(\hat{p}_i\right) = \frac{\sqrt{f_i}}{f_i} = \frac{1}{\sqrt{f_i}} \tag{5.2}$$

[4]The word "scamels" in the Shakespeare quote introducing this chapter is a classic hapax legomenon. Shakespeare only used it once and there are no other recorded uses by contemporary writers. Its meaning and etymology are completely unknown, making it debatable whether or not it is a valid English word.

The standard error for $\hat{\tilde{\lambda}}_1$ is $\lg\left(1 - \frac{1}{\sqrt{f_1}}\right)$ bits. The most common password in the YAHOO data set has $\hat{p}_1 \approx 1.08\%$ in our data set, giving the following estimates for stderr $\left(\hat{\tilde{\lambda}}_1\right)$:

M	69M	10M	1M	100k	10k	1k
stderr $\left(\hat{\tilde{\lambda}}_1\right)$	0.0016	0.0044	0.139	0.0446	0.1460	0.5234

Thus naive estimates for $\tilde{\lambda}_1$ are acceptably accurate even for very low M, as observed in Figure 5.1 where $\tilde{\lambda}_1$ was consistently estimated for small subsamples of the YAHOO data set. This argument can be extended for $\hat{\tilde{\lambda}}_\beta$ for small values of β by summing the standard error for the first β observations. In practice, we'll use naive estimates $\hat{\tilde{\lambda}}_\beta$ without further mention when M is large enough to ensure expected errors of less than 0.1 bits, which will hold for most of our sampled password data sets.

5.4 Good-Turing estimation of probabilities

The analysis of Good and Turing [125] produces an improved estimate for p_i:

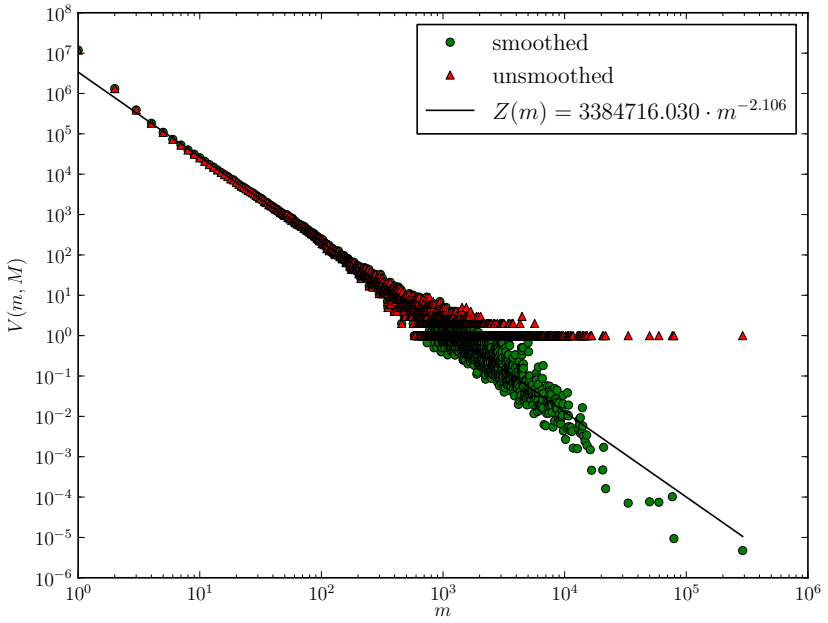
$$p_i^{\mathrm{GT}} = \frac{f_i + 1}{M} \cdot \frac{V(f_i + 1, M)}{V(f_i, M)} \tag{5.3}$$

A technical derivation is given by Baayen [23]. The naive estimate $\hat{p}_i$ is consistent in that, across all random subsamples including those in which x_i is not observed, the mean value of $\hat{p}_i$ is the true value p_i. This necessarily means over-estimating the probability of observed events since all events unobserved in a sample have $\hat{p}_i = 0$. In contrast, p_i^{GT} is consistent across all random subsamples conditioned on x_i being observed. This means that the sum $\sum_i p_i^{\mathrm{GT}} < 1$ since $\sum_i p_i^{\mathrm{GT}} = 0$ for unobserved events as well.

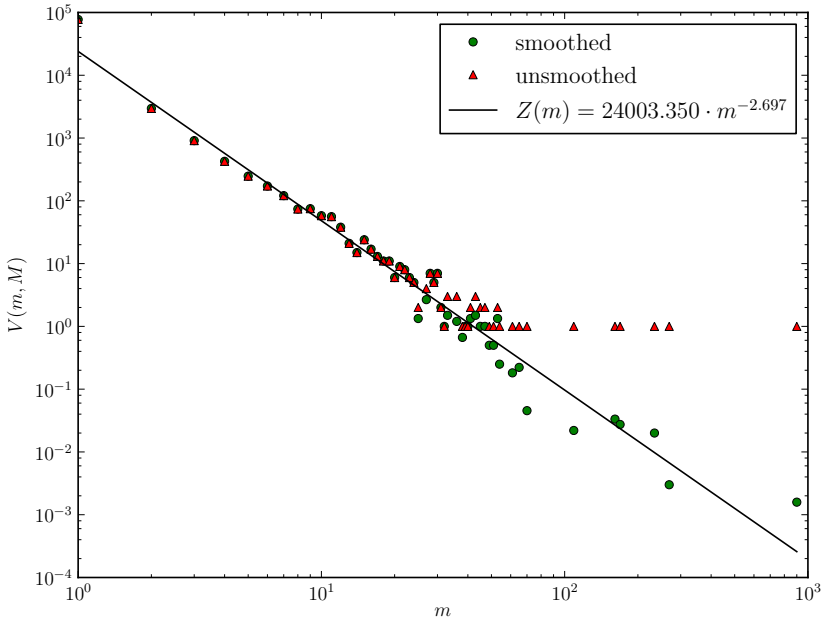
In effect, this formula leaves the estimates largely intact for higher-frequency events but heavily discounts the estimated probability for low-frequency events. We'll explore this using the ROCKYOU data set, which provides a more colourful example as plaintext passwords are available. The hapax legomena in this set would have their probability scaled by a factor of $\frac{2 \cdot V(2,M)}{V(1,M)} \approx 0.22$ using the Good-Turing estimator.

A major challenge to using Good-Turing estimates is that Equation 5.3 fails for higher-frequency events because the estimates for $V(f_i, M)$ are not smooth. For example, `123456` is the most common password in the ROCKYOU data set, occurring 290,729 times. Since $V(290730, M) = 0$, the basic Good-Turing estimator will produce an estimated probability 0 for `123456`.

Gale and Sampson introduced a modification [117] dubbed Simple Good-Turing which replaces the raw $V(m, M)$ values with smoothed values $Z(m)$. First, we scale down low values of



(a) Complete RockYou distribution ($M = 32,603,388$)



(b) Sub-sampled RockYou distribution ($M = 100,000$)

Figure 5.2: The plot of $V(m, M)$ vs. m for the full RockYou distribution and a subsample with $M = 100,000$. The unsmoothed plot (triangles) shows the raw counts, which cannot fall below 1. The smoothed plot, using the formula provided in [117], enables a plausible linear fit in both cases. However, the slope of the linear fit changes drastically in the sampled version.

x	f_x	f_x^{SGT}	$100 \cdot \hat{p}_x$	$100 \cdot p_x^{\mathrm{SGT}}$
123456	290729	290727.89	0.89171408	0.89171068
12345	79076	290727.89	0.24253921	0.89171068
password	59462	76787.89	0.18237982	0.23552121
rockyou	20901	21723.89	0.06410683	0.06663079
jessica	14103	14329.89	0.04325624	0.04395216
butterfly	10560	10729.89	0.03238927	0.03291037
charlie	7735	7764.89	0.02372453	0.02381622
diamond	5167	5173.89	0.01584805	0.01586919
freedom	3505	3512.89	0.01075042	0.01077463
letmein	2134	2136.89	0.00654533	0.00655421
bethany	1321	1320.90	0.00405173	0.00405140
lovers1	739	737.90	0.00226664	0.00226325
samanta	389	388.90	0.00119313	0.00119281
123456p	207	205.90	0.00063490	0.00063153
diving	111	109.90	0.00034046	0.00033710
flower23	63	61.91	0.00019323	0.00018990
scotty2hotty	34	32.93	0.00010428	0.00010099
lilballa	18	16.96	0.00005521	0.00005201
robbies	9	8.01	0.00002760	0.00002457
DANELLE	5	4.01	0.00001534	0.00001230
antanddeck06	3	1.88	0.00000920	0.00000577
babies8	2	0.88	0.00000613	0.00000271
sapo26	1	0.22	0.00000307	0.00000068

Table 5.1: Probability estimates for observed passwords in the RockYou distribution using maximum-likelihood estimation and Simple Good-Turing probability estimation.

$V(m, M)$, as all observed values of $V(m, M)$ are greater than 1 but the expected values $E[V(m, M)]$ for high m are much less than 1. Gale's smoothing formula is as follows, with m^+ and m^- representing the next-largest and next-smallest values of m for which $V(m, M) > 0$:

$$V_r(m, M) = \begin{cases} V(1, M) & \text{if } m = 1 \\ \frac{2 \cdot V(m,M)}{m^+ - m^-} & \text{if } 1 < m < \max(m) \\ \frac{2 \cdot V(m,M)}{2m - m^-} & \text{if } m = \max(m) \end{cases} \tag{5.4}$$

This is a simple heuristic formula which effectively spreads observed values of $V(m, M)$ across adjacent values of m for which $V(m, M) = 0$. We then fit the values $V_r(m, M)$ to a Zipf distribution $Z(m)$, as shown in Figure 5.2 for the RockYou distribution. We then replace the

raw counts $V(m, M)$ from Equation 5.3 with $Z(m)$, giving the Simple Good-Turing formula:

$$p_i^{\text{SGT}} = \frac{f_i + 1}{M} \cdot \frac{Z(f_i + 1)}{Z(f_i)} \tag{5.5}$$

Gale and Sampson recommend using p_i^{GT} for low-frequency events (since $V(m, M)$ will be close to $E[V(m, M)]$ for low m) and switching to p_i^{SGT} for higher-frequency events, using an automated test of the expected error in $V(m, M)$ to determine when to switch [117]. Example values of p_i^{SGT} and $\hat{p}_i$ for the RockYou distribution are provided in Table 5.1. In general, the estimates are nearly identical for high-frequency events and the Good-Turing estimates are much lower for low-frequency events.

5.5 The region of stability for aggregate metrics

We now turn to estimating $\tilde{\mu}_\alpha$ and $\tilde{G}_\alpha$. As discussed in §5.2, this will be impossible as $\alpha \to 1$ with sample size $M < O(N)$ due to the Low-Frequency Blindness Theorem. The Canonical Testing Theorem does inform us that we can accurately approximate $\tilde{\mu}_\alpha$ and $\tilde{G}_\alpha$ for some $\alpha < 1$ simply by ignoring low-frequency events in the sample, but the results are asymptotic and don't answer the practical question of which events occur frequently enough for us to use in producing an estimate. We'll consider the problem with respect to $\tilde{\mu}_\alpha$ in this section, since for the same value of α it is harder to estimate than $\tilde{G}_\alpha$ which relies strictly less on lower-frequency events.

Returning to the Yahoo data set, we plot the naive estimate $\hat{\tilde{\mu}}_\alpha$ in Figure 5.3 at several sample sizes. We observe two important facts. First, estimates are very consistent at low α even for dramatically smaller subsamples, giving us hope that we can estimate $\tilde{\mu}_\alpha$ accurately in some useful cases. Second, by computing $\hat{\tilde{\mu}}_\alpha$ for many random subsamples we see that the estimates don't vary significantly in different random subsamples. Thus, we lose very little precision in estimating $\tilde{\mu}_\alpha$ using a random subsample, it is only the underestimation bias for higher α that is of practical concern.

The reason for the consistency of naive estimates is that, for large enough sample size M, the frequency counts $V(m, M)$ will be based on a large number of events for low m. Even though the observed frequency of individual events vary considerably, the aggregate $V(m, M)$ are very consistent for a fixed M. However, each count $V(m, M)$ will consistently tend to be too large. For our naive estimation $\hat{\tilde{\mu}}_\alpha$ to be accurate, we would need to have each $V(m, M)$ be an unbiased estimate of the number of events with probability $\approx \frac{m}{M}$:

$$V^*(m, M) = M \cdot \sum p_i : \frac{m - \frac{1}{2}}{M} \leq p_i \leq \frac{m + \frac{1}{2}}{M} \tag{5.6}$$

Of the events observed m times, many more will have a true probability $p_i < \frac{m}{M}$ than $p_i > \frac{m}{M}$ simply because there are more lower-frequency events to be over-represented than the inverse. Yet naive estimation considers all events with $f_i = m$ to have $\hat{p}_i = \frac{m}{M}$ when computing $\hat{\tilde{\mu}}_\alpha$.

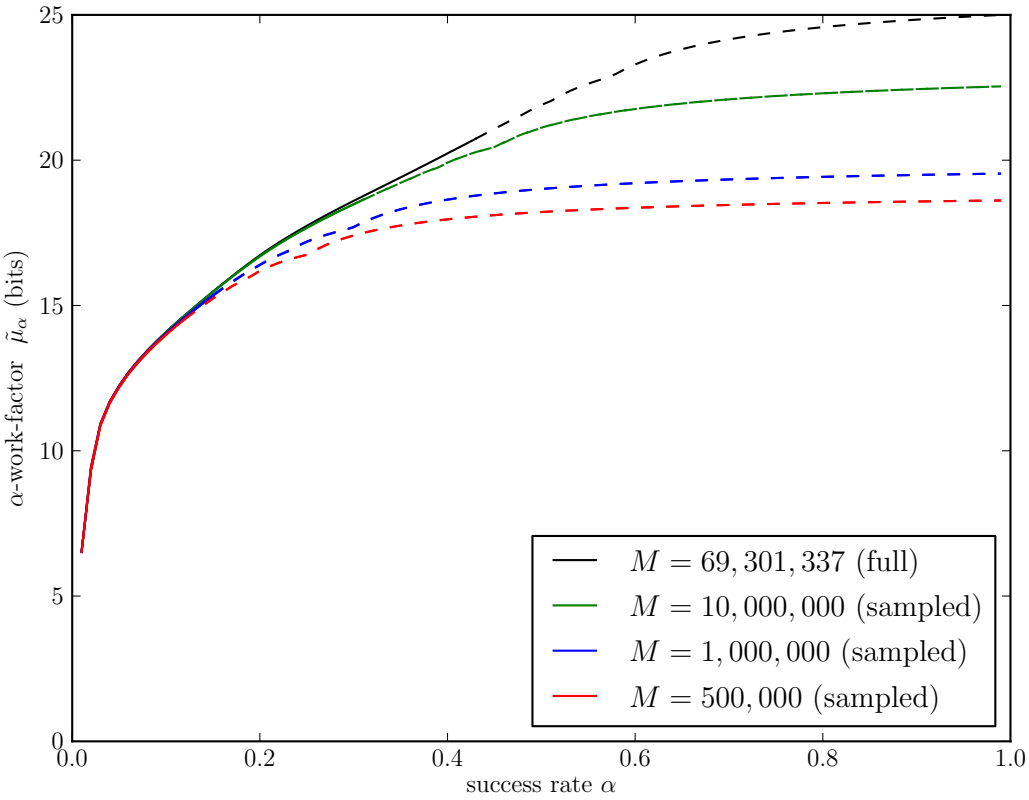


Figure 5.3: Estimated guessing curves with reduced sample size M. Subsamples were computed randomly without replacement, to simulate having stopped collecting samples earlier. Each line is broken after the calculated maximum confidence interval α_* for an accuracy of 0.5 bits; the subsamples agree very closely below this point. Each subsample is actually plotted twice, representing the 1st and 99th percentiles from 1,000 random samples. These are almost imperceptible, indicating that randomness in sampling is not a major factor compared to underestimation bias.

With perfect knowledge of the underlying distribution $\mathcal{X}$, we could approximate[5] the expected value $\mathrm{E}[V(m, M)]$ of each count quite accurately by modeling the number of observations of each event as an independent binomial variable and summing the expected probability of each event being observed m times:

$$\mathrm{E}\left[V(m, M)\right] = \sum_{1 \le i \le N} \binom{M}{m} p_i^m (1 - p_i)^{(M-m)} \tag{5.7}$$

We could then use this to calculate the expected ratio by which our estimates for the low-frequency counts $V(m, M)$ are too high:

$$\mathrm{bias}(V(m, M)) = \frac{\mathrm{E}\left[V(m, M)\right]}{V^*(m, M)} \tag{5.8}$$

[5]Computing the expected value precisely would require calculating the multinomial probabilities of all possible outcomes of M draws from $\mathcal{X}$, which is computationally intractable.

In practice, we don't have perfect knowledge of the underlying distribution, making it impossible for us to apply Equation 5.8 directly. We can instead obtain an upper-bound through a statistical technique called *bootstrapping* [95]. The general idea of bootstrapping is to take many resamples of size M, with replacement, from our sampled distribution $\hat{\mathcal{X}}$ in order to examine the distribution of values for the statistic of interest, in our case $V(m, M)$. In each bootstrap resample, we expect $V^{\text{bs}}(m, M)$ to be an overestimate of $V(m, M)$ from the original sample, just as $V(m, M)$ was itself an overestimate of $V^*(m, M)$. Because the sampled distribution $\hat{\mathcal{X}}$ will be more skewed than the underlying distribution $\mathcal{X}$, this gives us an approximation for the error bound from Equation 5.8:

$$\frac{\mathrm{E}\left[V(m, M)\right]}{V^*(m, M)} \lessapprox \frac{\mathrm{E}\left[V^{\text{bs}}(m, M)\right]}{\mathrm{E}\left[V(m, M)\right]} \tag{5.9}$$

We compare results provided by this technique to actual counts in Table 5.2. We use the full YAHOO data set as an approximation for the underlying distribution, take random samples of size $M = 1\text{M}$ and then take bootstrap resamples to estimate the error in each of the subsamples. Note that this breaks down for $m = 1$, because the number of events $V^*(m, M)$ which have probability $\leq \frac{1.5}{M}$ which would appear in an ideal sample is many times greater than M itself.[6] For $m > 1$, the simple estimates $V(m, M)$ very quickly become good approximations for $V^*(m, M)$, and hence can be used to estimate guessing statistics. Equation 5.9 provides a loose bound on the true bias for low m and become progressively tighter.

The naive estimate $\hat{\tilde{\mu}}_\alpha$ will underestimate the true value of $\tilde{\mu}_\alpha$ because the counts $V(m, M)$ tend to be slight overestimates, meaning we underestimate the number of events μ needed to have a cumulative probability of success α. Denoting α_m for the value of α which only requires guessing events of frequency $\geq m$ in our sample, we can estimate the total overestimation bias in $\hat{\tilde{\mu}}_{\alpha_m}$ by taking a weighted sum of the bias in each of the components $V(m, M)$:

$$\text{bias}^{\text{bs}}\left(\hat{\tilde{\mu}}_{\alpha_m}\right) \lessapprox \frac{1}{\alpha_m \cdot M} \sum_{m' \geq m} m' \cdot \frac{\bar{V^{\text{bs}}}(m', M)}{V(m', M)} \tag{5.10}$$

In Table 5.2 we demonstrate the usefulness of Equation 5.10 as an upper bound on the true bias of $\hat{\tilde{\mu}}_\alpha$. As m increases, $\hat{\tilde{\mu}}_{\alpha_m}$ quickly becomes a very good estimate. Not listed in Table 5.2 is the standard error of $\hat{\tilde{\mu}}_\alpha$ across random subsamples, because this is extremely low (< 0.01 bits) and is dwarfed by the systematic bias.

We'll choose 0.5 bits as a desired accuracy level for $\hat{\tilde{\mu}}_\alpha$ and use our bootstrapping method to estimate m^* for which the bias will be less than 0.5 bits. We then call $\alpha_* = \alpha_{m^*}$ the limit below which we'll accept $\hat{\tilde{\mu}}_\alpha$ as a reasonable estimate. As seen in Figure 5.1 and 5.3, this method accurately predicts the point at which $\hat{\tilde{\mu}}_\alpha$ begins to diverge from the true value. For our full YAHOO data set, we have $\alpha_* \approx 0.44$.

[6] This problem is yet another consequence of the fact that many events go unobserved in random subsamples.

m	$V^*(m,M)$	$\bar{V}(m,M)$	$\bar{V^{\text{bs}}}(m,M)$	$\frac{\bar{V}(m,M)}{V^*(m,M)}$	$\frac{\bar{V^{\text{bs}}}(m,M)}{V(m,M)}$	$\bar{\alpha}_m$	$\text{bias}^{\text{bs}}\left(\hat{\tilde{\mu}}_{\alpha_m}\right)$	$\frac{\tilde{\mu}_{\alpha_m}(\mathcal{X})}{\tilde{\mu}_{\alpha_m}(\hat{\mathcal{X}})}$
2	18392	34384.3	142020.6	1.87	4.13	0.30	3.77	1.88
3	6244	10795.9	53275.0	1.73	4.94	0.23	3.29	1.41
4	3096	4894.4	17512.3	1.58	3.58	0.20	2.18	1.26
5	1870	2644.0	6265.6	1.41	2.37	0.18	1.56	1.18
6	1330	1663.9	2861.5	1.25	1.72	0.16	1.31	1.13
7	926	1098.6	1646.0	1.19	1.50	0.15	1.21	1.11
8	668	815.3	1087.8	1.22	1.33	0.15	1.15	1.09
9	506	616.3	778.1	1.22	1.27	0.14	1.12	1.08
10	424	463.3	586.4	1.09	1.27	0.13	1.10	1.07
11	349	387.7	459.4	1.11	1.19	0.13	1.08	1.07
12	308	326.1	370.9	1.06	1.14	0.12	1.07	1.06
13	255	274.8	306.6	1.08	1.12	0.12	1.06	1.06
14	226	239.8	258.2	1.06	1.08	0.12	1.05	1.05
15	185	198.3	220.7	1.07	1.12	0.11	1.05	1.05
16	151	178.8	191.0	1.18	1.07	0.11	1.05	1.05
17	144	154.3	166.8	1.07	1.09	0.11	1.04	1.04
18	147	135.8	146.9	0.92	1.09	0.11	1.04	1.04
19	98	119.5	130.3	1.22	1.10	0.10	1.04	1.04
20	109	116.2	116.2	1.07	1.00	0.10	1.04	1.04

Table 5.2: Ideal and observed frequency counts for low m in the YAHOO data set for 1,000 random samples with $M = 1\text{M}$. The "ideal" value $V^*(m,M)$ value is approximated using Equation 5.6 for the full data set ($M = 69\text{M}$), while the bootstrap prediction $\bar{V}^{\text{bs}}(m,M)$ is computed by randomly resampling M times from each subsampled distribution. The bias predicted by the bootstrap method as described in Equation 5.10, denoted bias^{bs}, is compared to the actual overestimation of $\tilde{\mu}_{\alpha_m}$ using sampled data for α_m, the cumulative probability of events observed at least m times.

5.6 Parametric extension of our approximations

Estimating $\tilde{\mu}_\alpha$ for $\alpha > \alpha_*$ requires making assumptions about the structure of the underlying distribution. We'll assume the distribution of human-memorable secrets might be related to the distribution of words in natural language, for which many probabilistic models have been proposed [23]. Perhaps the most-frequently conjectured model is a *power-law* distribution:[7]

$$\Pr[p_x > y] \propto y^{1-a} \tag{5.11}$$

Unfortunately, using a power-law distribution to estimate $\tilde{\mu}_\alpha$ is problematic for two reasons. First, estimates for the parameter a are known to decrease significantly with sample size [23]. Using maximum-likelihood fitting techniques [69] for the observed counts in the YAHOO data set we get the following estimates for a at different sample sizes:

M	69M	10M	1M	100k
$\hat{a}$	2.99	3.23	3.70	4.21

The second problem is this model fits our observed counts, which are integers greater than 1. To correctly estimate $\tilde{\mu}_\alpha$ from a sample, we need to model the presence of events for which $p_x \cdot M < 1$. Fitting a power law distribution to observed counts requires implicitly assuming a non-zero minimum password probability [69], which there is no meaningful way of doing.

Instead we must choose a continuous distribution Ψ to model the distribution of events' probabilities and not their observed counts, which is the approach taken by Font et al. [106] for word frequencies. We model the probability of observing an event k times using a mixture model: first we draw a probability p randomly according to the probability density function $\psi(p)$, then we draw from a Poisson distribution with expectation $p \cdot M$ to model the number of times we observe this event in the sample:

$$\Pr[k \text{ observations}] = \frac{\int_0^1 \frac{(p\cdot M)^k \cdot e^{-p\cdot M}}{k!} \psi(p) \mathrm{d}p}{1 - \int_0^1 e^{-p\cdot M} \psi(p) \mathrm{d}p} \tag{5.12}$$

The numerator integrates the possibility of seeing an event with probability p exactly k times in a sample, weighted by the probability $\psi(p)$ of seeing an event with probability p. The denominator corrects for the probability of not seeing some passwords in a sample. This formulation allows us to take a set of counts from a sample $\{f_1, f_2, \ldots\}$ and find the parameters for $\psi(p)$ which maximise the likelihood of our observed sample:

$$\text{Likelihood} = \prod_{i=1}^{\hat{N}} \Pr[f_i \text{ observations}] \tag{5.13}$$

[7] Power-law distributions are also called Pareto or Zipfian distributions, which can all be shown to be equivalent formulations [23].

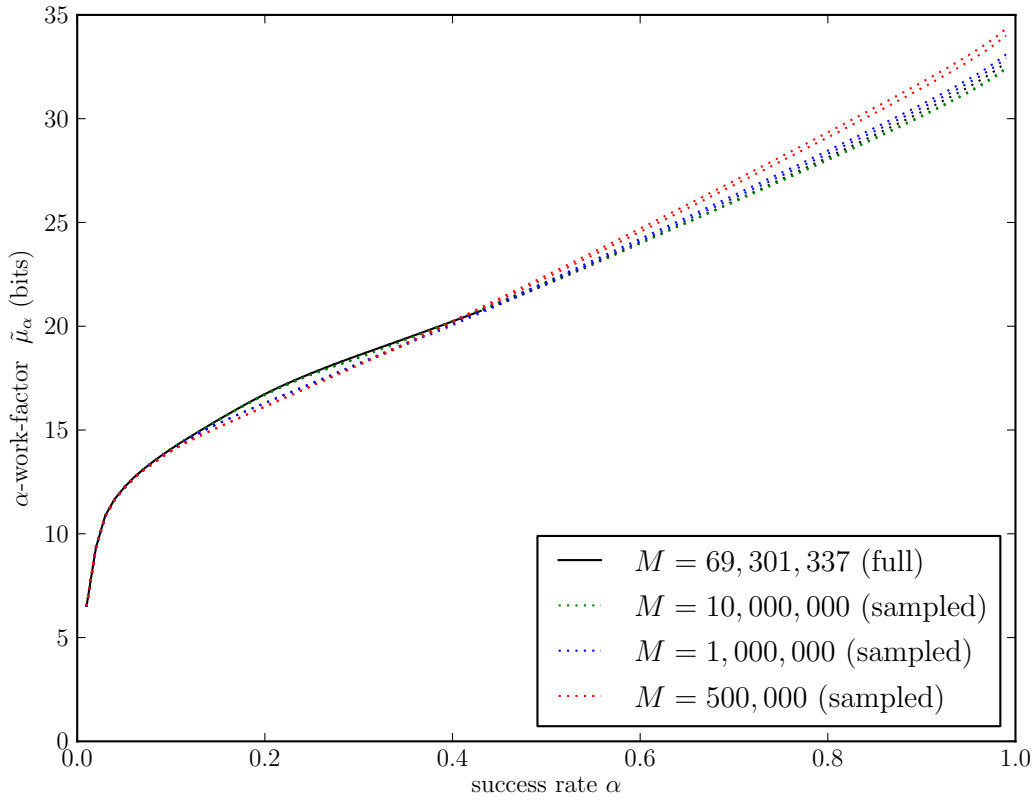


Figure 5.4: Extrapolated estimates for $\tilde{\mu}_\alpha$ using the zero-truncated Sichel/Poisson distribution, as proposed for word frequency modeling [106]. Compared to naive estimates (Figure 5.3) we are able to mitigate much of the bias due to sample size. Each plot shows the 99% confidence interval from 1,000 random subsamples. Error from lack of fit of the model is much greater than the standard deviation of each sample.

This model has been effectively applied to word frequencies using the *generalised inverse-Gaussian* or *Sichel* distribution:[8]

$$\psi(p|b,c,g) = \frac{2^{g-1}p^{g-1}e^{\frac{p}{c}-\frac{b^2c}{4p}}}{(bc)^g \cdot K_g(b)} \tag{5.14}$$

where K_g is the modified Bessel function of the second kind [274].

The Sichel distribution is useful because it blends power-law $\left(p^{g-1}\right)$ and exponential $\left(e^{\frac{p}{c}-\frac{b^2c}{4p}}\right)$ behaviour and produces a well-formed probability distribution. As proved by Font et al. [106], by plugging Equation 5.14 into Equation 5.12 for ψ and solving the integral, we obtain:

$$\Pr[k|b,c,g] = \frac{(\frac{1}{2}\cdot\frac{bcn}{\sqrt{1+cn}})^r \cdot K_{r+g}(b\sqrt{1+cn})}{r!\left((1+cn)^{\frac{g}{2}}K_g(b) - K_g(b\sqrt{1+cn})\right)} \tag{5.15}$$

[8]The generalised inverse Gaussian distribution is often called the Sichel distribution after its initial use by Herbert Sichel in 1975 to model word frequencies [274].

	69M	10M	1M	500k
$\hat{\tilde{\mu}}_{0.25}$	17.74	17.67–17.67	17.24–17.25	17.07–17.17
$\hat{\tilde{\mu}}_{0.5}$	22.01	22.09–22.11	22.06–22.11	22.28–22.48
$\hat{\tilde{\mu}}_{0.75}$	27.07	26.98-27.01	27.25–27.35	27.02–27.89

Table 5.3: 99% confidence intervals for extrapolated estimates of $\tilde{\mu}_\alpha$ for the Yahoo data set.

We can compute Equation 5.13 using Equation 5.14 for different parameters of b, c, g. Fortunately, for $b > 0, c > 0, g < 0$ there is only one maximum of this function [106], which enables approximation of the maximum-likelihood fit efficiently by gradient descent.

This combined model is called the *zero-truncated generalised inverse-Gaussian/Poisson* distribution [106]. We can combine this model with our observed data in the well-approximated region to produced an "extrapolated distribution" which will better approximate $\tilde{\mu}_\alpha$.

We remove all observed events with $f_i < m_*$ to leave the well-approximated region of the distribution unchanged and add synthetic events according to our estimated model $\psi(p)$. This is achieved by dividing the region $\left(0, \frac{m_*}{M}\right)$ into discrete bins, with increasingly small bins near the value p^+ which maximises $\psi(p^+)$. Into each bin (p_j, p_{j+1}) we insert $\hat{N} \cdot \int_{p_j}^{p_{j+1}} \psi(p)\mathrm{d}p$ events of observed frequency $\frac{p_j + p_{j+1}}{2 \cdot M}$. We then normalise the probability of all synthetic events by multiplying the correction factor $\frac{1}{\alpha_*} \cdot \int_{\frac{m_*}{M}}^{1} \psi(p)\mathrm{d}p$ to leave the head of the distribution intact.

We cannot conclude that this model fully explains password selection; using a Kolmogorov-Smirnov test we can reject with very high confidence ($p > 0.99$) the hypothesis that the Yahoo sample was drawn from the modeled distribution. However, this model enables practical estimation of $\tilde{\mu}_\alpha$ from subsampled distributions for a much larger interval of α, as shown in Figure 5.4. For the Yahoo data set we can establish confidence intervals by taking many random subsamples and using the 1^{st} and 99^{th} percentile estimates, as listed in Table 5.3.

We can conclude that any demographic subsample which produces estimates outside of this interval is significantly different from the general population of users with confidence $p > 0.99$. In §6, we will use this projection technique to estimate $\tilde{\mu}_\alpha$ for demographic subsamples of the Yahoo data set, noting our empirical confidence interval where appropriate.

We note several points of caution for using extrapolated estimates. We can see in Table 5.3 that the extrapolated estimates are biased towards under-correction for lower values of α and over-correction for higher values of α. The model also fails to capture the observed phenomenon of strong pseudorandom passwords in the tail of the distribution, which would produce estimates well over 100 bits as $\alpha \to 1$ and have no analog in natural language. In general, we have little understanding of the password distribution for $\alpha > 0.5$, for which cracking data (§2.5.1) is sparse. When we need to use extrapolated estimates it is prudent to only use them for $\alpha \lessapprox 0.5$.

There is a strong tendency for people to choose relatively short and simple passwords that they can remember.

—Robert H. Morris and Ken Thompson, 1979 [212]

Chapter 6

Guessing difficulty of passwords

We are now ready to analyse the guessing difficulty of text passwords. Our primary data source is a corpus of almost 70 M passwords collected at Yahoo!; we'll first discuss the privacy-preserving experimental set-up which produced this data set and then analyse this data and compare it to other available password data.

6.1 Anonymised data collection

In small-scale studies, users may be willing to share passwords with researchers under ethics oversight [173, 169]. This approach will never be practical for large-scale databases with millions of passwords, as are required for statistical analysis to be useful. Because purely statistical password metrics can be computed without access to passwords in plaintext form, it is possible to collect a large sample from real users.

6.1.1 Cleartext passwords and the Domesday attack

Collecting *only* passwords, with no additional identifying information, can superficially ameliorate privacy concerns. An example is the RockYou data set (§D), which was released as a list only of passwords with no corresponding account information.[1] Unfortunately, there remains an unacceptable risk from releasing such a data set: every released password can be included in password dictionaries and used in future password cracking attacks with priority over passwords purely generated by mangling rules.

Suppose Alice has taken the time to memorise a truly random 12-character alphanumeric password. Since the space of such passwords is $2^{71.5}$, initially her password can be considered

[1] The anonymous individual who released the data stated that he or she stripped user accounts from the database to limit the potential for abuse.

safe even against offline cracking attacks. If Alice's password is inadvertently leaked within a large list of passwords, it may now be prioritised by password cracking software on the basis that it has been used by at least one human once. We propose calling this a *Domesday attack* after the historical attempt to record all property ownership in England in one book. The utility of this attack depends on the total number of passwords ever used by humans (N_{global}) being considerably less than the space searchable by offline password cracking tools. We can make a rough estimate of the Domesday password space as:

$$N_{\text{global}} \approx \text{population} \cdot \frac{\#\text{passwords}}{\text{person}} \cdot (1 - \text{re-use rate})$$

The estimated human population of the Earth surpassed 7 billion $\approx 2^{32.7}$ in 2012, which serves as an upper bound on the population of people who have ever used a password. The most reliable large-scale user study, of Flôrencio and Herley [103] in 2007, found the average web user maintains 6.5 different passwords. Finally, in the RockYou data set, 56% of observed passwords were repeats, likely an under-estimate of the global rate.

Thus, we can estimate that $N_{\text{global}} \lessapprox 2^{34.3}$. This quantity is already cheap to store and is far smaller than what can be produced dynamically by modern password-cracking tools in an offline search. This suggests that a Domesday dictionary of passwords would significantly increase efficiency over brute-force password cracking.

Thus, leaking Alice's password only once may make it vulnerable in perpetuity to a Domesday attack where it was previously safe against brute force. We expect such attacks to be increasingly practical as more large-scale password leaks occur, especially as the world population and human memory constraints are growing much more slowly than storage and processing capability. We therefore reject the appropriateness of ever collecting cleartext user passwords, with or without additional identifying information.

6.1.2 Isomorphic distributions

When estimating from a random sample (§5), the only requirement when a password $\mathbf{x}$ is observed is that we can tell if $\mathbf{x}$ was observed before. This can be done equally well if we relabel every password $\mathbf{x} \to f(\mathbf{x})$ using any deterministic, collision-free function f, producing an *isomorphic distribution* with the same values for guessing metrics.

Because passwords are typically[2] sent in plaintext form to a website's login server, we can set up a collection experiment in which all submitted passwords are masked using f before being added to our data set. Our goal is to design a collection experiment enabling data to be collected and shared with independent researchers with minimal security risk.

[2]It is possible for a site to request the user's browser hash passwords prior to submitting them [42], though this is very rare in practice [47].

Masking with a cryptographic hash function

To preserve privacy, we want to use a masking function f which is *one-way*, that is, for which there is no known method for calculating f^{-1} or computing any information about $\mathbf{x}$ given $f(\mathbf{x})$. A function which is one-way, deterministic and collision-resistant[3] is called a *cryptographic hash function* [22], often denoted $\mathbf{H}$. The best-known algorithms (though not the most secure[4]) are MD5 and SHA-1, which are frequently used in hashing stored passwords to mitigate database compromises (§2.2.1).

Collecting passwords which are masked with a cryptographic hash does not appreciably contribute to a Domesday attack because any passwords revealed to be in the data set must have already been within the range searchable by password-cracking tools. Thus, adding them to a Domesday dictionary will not make any previously-invulnerable passwords vulnerable.

However, it is still possible to carry out an offline brute-force attack to try to discover passwords in the data set. Therefore it would not be acceptable to collect such a data set along with any identifying information attached to individual passwords. This is particularly important because statistical analysis will not work if passwords are salted prior to hashing (§2.2.1), which would undermine the requirement of f being deterministic and occlude frequency counts of repeated passwords. Without salting, the collected data set would be vulnerable to a parallel dictionary attack or a precomputed rainbow table if a common hash function were used.

Masking with a secret one-way function

The core problem with a deterministic one-way function is that a malicious party can evaluate f on trial passwords and test for their membership in a published data set. This can be prevented if f is a secret function which cannot be publicly computed. One example[5] is a keyed hash function, $f(\mathbf{x}) = \mathbf{H}(\mathbf{x}||r)$ for some cryptographic hash function $\mathbf{H}$ and secret nonce[6] r. The inclusion of the secret value r means that a malicious party can't evaluate f, so access to the data set masked with f will not enable brute-force attacks.

To use a secret one-way function in this manner, our collection server must generate a random nonce r at the beginning of the experiment which is so large it can never be found through brute force. 128 bits is a conservative choice. Every time the server sees a user $\mathbf{i}$ log in with a password x, it computes $f(\mathbf{x}) = \mathbf{H}(\mathbf{x}||r)$ and adds the resulting value to the data set. The

[3] We assumed in our definition that f was a collision-free function. Hash functions cannot be collision-free due to their infinite input space and finite output space, but finding collisions is a computationally intractable problem. Thus, the risk of hash collisions between actually used passwords is negligible.

[4] Both MD5 and SHA-1 are vulnerable to collision-finding attacks [304] though the one-wayness of neither algorithm has been compromised.

[5] There are many other possible secret one-way functions, notably any cryptographic MAC function [22].

[6] The secret value can be thought of as a cryptographic key in terms of the security it provides by resisting a brute-force attack, but since it will never be used for encryption or decryption nonce is a more accurate term.

collection server must then delete r from its memory prior to releasing the data for research, preventing any possible brute-force attacks.

6.1.3 Preventing duplicates from returning users

In collecting a masked distribution, we want to study the distribution of passwords across users and not individual login attempts. This requires that during the collection experiment when our server sees the login credential $(\mathbf{i}, \mathbf{x})$ it can check to see if $\mathbf{i}$ has already logged in during the experiment. One solution is to simply add the value $(\mathbf{i}, f(\mathbf{x}))$ to the data set, enabling filtering of duplicate items later.

This approach would enable *linking attacks*, however, in which the data set reveals if two users $\mathbf{i}_1, \mathbf{i}_2$ use the same password (without revealing what that password is). Linking two accounts means that compromising either one can compromise the other as well[7]. If a large portion of accounts are de-anonymised, this potentially allows an attacker to leverage a small number of known passwords into a large number of compromised accounts.

There is also the possibility of active attacks, in which an attacker creates a large number of accounts prior to the experiment, all with different known passwords. If an attacker can create many more accounts than the online guessing limit imposed by the normal login server, then this may be a better attack vector than online guessing.

To prevent linking attacks, our server must not write user identifiers $\mathbf{i}$ directly but $f(\mathbf{i}) = \mathbf{H}(\mathbf{i}||r)$ instead, which will prevent inference of $\mathbf{i}$ just as it prevents inferring passwords. A more efficient solution to prevent writing duplicate values to the data set is to maintain a *Bloom filter* of seen identities $f(\mathbf{i})$, a special data structure which allows identifying repeated values in a stream with very compact storage [39].

6.1.4 Collecting demographic data

It is desirable to collect demographic data along with passwords during our experiment to enable more detailed analysis of password choices. A simplistic solution would be to store a tuple containing demographic details of interest along with masked passwords:

$$\{f(\mathbf{x}), \text{age} = 27, \text{language} = \text{en}, \text{location} = \text{UK} \dots\}$$

This approach carries the risk that the user $\mathbf{i}$ can be *re-identified* by the uniqueness of his or her demographic information [83], enabling linking attacks as described above. Re-identification attacks can be surprisingly effective. An investigation by Sweeney in 2000 demonstrated that the combination of postcode, gender, and date of birth is unique for 87% of the US population [288]. Narayanan et al. demonstrated a practical re-identification attack against a data

[7] There are many ways an attacker can compromise a user's password, especially if it can be attacked at other sites where it is registered [152].

set of movie reviews superficially anonymised for research purposes in 2006 [219]. It would almost certainly be possible to re-identify most users given the number and detail of predicates we would like to study.

One approach is to prevent re-identification by perturbing the demographic details stored to prevent direct inference. The recent field of research on *differential privacy*, introduced by Dwork [94], studies how much modification must be made to each individual demographic item to ensure that it is computationally infeasible to detect the presence of any known individual in the data set. These techniques can significantly affect the utility of data, however, by requiring a large amount of perturbation to achieve guaranteed security.

A simpler approach in our case is *de-aggregation* by not storing large tuples of demographic details together. For each of ℓ demographic predicate functions $d_1, d_2, \ldots, d_\ell$ our collection server maintains separate histograms $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_\ell$. When the credential $(\mathbf{i}, \mathbf{x})$ is observed, if $f(\mathbf{i})$ has not yet been observed the collection server adds $f(\mathbf{x})$ to every histogram $\mathcal{H}_n$ for which $d_n(\mathbf{i})$ is true.

At the end of the experiment, the server can delete any histograms for which fewer than k passwords were observed. This will preserve k-anonymity in the published data set, as originally defined by Sweeney [289], meaning that no data will be released identifying any group of users smaller than k. For our purposes, small histograms ($k \lesssim 10,000$) will not enable computation of interesting guessing metrics, so we lose very little by removing them from consideration. Pseudocode for our complete collection set-up is provided in Figure 6.1.

6.1.5 Deployment

We implemented the experimental collection server described above in cooperation with Yahoo!, a large online website which provides email, news, social networking, and many other services. Yahoo! maintains a single login system for all of its web properties and has hundreds of millions of active users. The audited collection code consisted of a few dozens lines of Perl. The secret r was generated by combining a secret seed provided by a Yahoo! manager and locally generated entropy. The server monitored login traffic in front of a random subset of Yahoo! login servers for a 48-hour period from May 23–25, 2011, observing 69,301,337 unique users and constructing separate histograms for 328 different predicate functions. Of these, those which saw fewer than $k = 10,000$ samples were discarded.

6.2 Analysis of Yahoo! data

We now analyse the collected YAHOO data. Of the 328 subpopulations for which we compiled separate distributions, we summarise the most interesting in Table 6.1. Estimates in italics are not significantly different from the aggregate population of users, as discussed in §5.

```
function PASSWORD_COLLECTION_EXPERIMENT(d, k)
    r ← generate_random_bits(128)
    B ← bloom_filter()
    for 1 ≤ j ≤ len(d) do
        H[j] ← histogram()
    while (i, x) ← read_next_credential() do
        if hash(r, i) ∉ B then
            B.add(hash(r, i))
            for 1 ≤ j ≤ len(d) do
                if d[j](i) then
                    H[i].add(hash(r, x))
    for 1 ≤ j ≤ len(d) do
        if len(H[i]) < k then
            delete H[j]
    delete r
    delete B
    return H
```

Figure 6.1: Pseudocode for a privacy-preserving password collection proxy server.

All sub-distributions had similar guessing metrics: the range of $\tilde{\lambda}_1$ was 5.0–9.1 bits and $\tilde{\lambda}_{10}$ from 7.5–10.9 bits, just over one decimal order of magnitude in variation. Variation in $\tilde{G}_{0.5}$ was substantially larger, with the weakest population having an estimated 17.0 bits and the strongest 26.6 (nearly three decimal orders of magnitude). While the absolute differences are relatively small and there is no notably "good" population of users which isn't generally vulnerable to guessing attacks, we can identify many groups which are statistically different from the overall population (§5).

Demographically, users' reported gender had a small but split effect, with male-chosen passwords being slightly more vulnerable to online attack and slightly stronger against offline attack. There is a general trend towards better password selection with users' age, particularly against online attacks, where password strength increases smoothly across different age groups by about 1 bit between the youngest users and the oldest users. Far more substantial were the effects of language: passwords chosen by Indonesian-speaking users were amongst the weakest subpopulations identified with $\tilde{\lambda}_1 \approx 5.5$ bits. In contrast, German- and Korean-speaking users provided relatively strong passwords with $\tilde{\lambda}_1 \approx 7.5$ bits.

Table 6.1: Key guessing statistics for passwords of various populations of Yahoo! users. Estimates in italics are not different from the overall population to a statistically significant degree, using the methodology discussed in §5.

	M	$\hat{\tilde{\lambda}}_1$	$\hat{\tilde{\lambda}}_{10}$	$\hat{\tilde{\lambda}}_{100}$	$\hat{\tilde{G}}_{0.25}$	$\hat{\tilde{G}}_{0.5}$
all passwords	**69301337**	**6.5**	**9.1**	**11.4**	**17.6**	**21.6**
gender (self-reported)						
female	30545765	6.9	9.3	*11.5*	*17.2*	21.1
male	38624554	6.3	8.8	*11.3*	17.7	*21.8*
age (self-reported)						
13–24	18199547	6.3	8.7	11.1	16.7	20.9
25–34	22380694	6.2	8.8	*11.2*	*17.1*	21.2
35–44	12983954	6.8	9.4	11.7	*17.4*	21.3
45–54	8075887	7.3	9.8	11.8	*17.3*	21.3
≥ 55	7110689	7.5	9.8	11.8	*17.3*	21.4
language preference						
Chinese	1564364	*6.5*	8.6	11.1	*17.3*	*22.0*
German	1127474	7.4	9.7	*11.3*	15.8	19.7
English	55805764	6.5	*9.0*	*11.3*	*17.4*	21.5
French	2084219	6.9	*9.0*	10.9	14.8	18.6
Indonesian	1061540	5.5	7.9	10.2	14.3	17.0
Italian	811133	6.8	*9.0*	10.7	14.5	18.0
Korean	530759	7.5	9.5	11.7	18.1	22.7
Portuguese	2060256	6.5	*9.0*	11.0	15.6	18.8
Spanish	3065901	6.6	*9.1*	11.0	15.6	19.7
tenure of account						
≤ 1 year	5182527	6.9	*9.1*	11.7	18.0	22.5
1–2 years	5182527	6.9	*9.1*	11.7	18.0	22.5
2–3 years	12261556	6.2	8.6	11.2	17.7	*21.8*
3–4 years	10332348	6.2	8.8	*11.3*	*17.5*	21.6
4–5 years	9290840	6.1	8.8	*11.2*	*17.2*	21.2
≥ 5 years	29104856	6.8	9.3	*11.5*	*17.2*	21.2
password requirements at registration						
none	20434875	6.6	9.2	*11.4*	16.8	20.7
6 char. minimum	13332334	6.5	*9.0*	*11.4*	*17.6*	*21.6*
last recorded login						
< 30 days	32627777	*6.5*	*9.0*	*11.4*	*17.5*	21.5
< 90 days	55777259	6.5	*9.0*	*11.4*	*17.5*	21.5

continued on next page ...

Table 6.1 — continued from previous page

	M	$\hat{\tilde{\lambda}}_1$	$\hat{\tilde{\lambda}}_{10}$	$\hat{\tilde{\lambda}}_{100}$	$\hat{\tilde{G}}_{0.25}$	$\hat{\tilde{G}}_{0.5}$
all passwords	**69301337**	**6.5**	**9.1**	**11.4**	**17.6**	**21.6**
$>$ 90 days	8212643	7.0	9.5	11.7	17.7	*21.9*
number of login locations						
1	16447906	6.0	8.6	11.2	*17.1*	21.1
$\geq$2	52853431	6.7	*9.2*	*11.5*	17.7	*21.7*
$\geq$ 10	17146723	7.3	9.7	11.8	18.3	22.6
number of password changes						
none	52117133	6.2	8.8	*11.2*	*17.1*	20.9
1	9608164	8.3	10.4	12.3	18.8	23.2
$>$1	7576040	8.6	10.7	12.5	19.5	24.2
$\geq$5	930035	9.1	10.9	12.7	19.7	25.9
number of password resets (forgotten password)						
none	61805038	6.4	8.9	*11.3*	*17.3*	21.3
1	4378667	8.2	10.5	12.5	19.2	23.8
$>$1	3117632	8.7	10.8	12.8	19.7	24.6
$\geq$5	387469	8.7	10.6	12.8	19.9	26.6
amount of data stored with Yahoo!						
1$^{\text{st}}$ quartile	9830792	5.6	8.2	10.8	*17.3*	21.5
2$^{\text{nd}}$ quartile	20702119	6.3	8.8	*11.3*	*17.5*	21.5
3$^{\text{rd}}$ quartile	21307618	6.8	9.3	*11.5*	*17.5*	21.4
4$^{\text{th}}$ quartile	17447029	7.6	10.0	11.9	17.8	*22.0*
usage of different Yahoo! features						
media sharing	5976663	7.7	10.1	12.0	18.0	*22.3*
retail	2139160	8.8	10.5	11.9	16.8	21.4
webmail	15965774	6.3	8.8	*11.3*	*17.4*	21.2
chat	37337890	6.2	8.7	11.2	*17.1*	21.2
social networking	14204900	7.1	9.6	11.7	17.7	*21.8*
mobile access	20676566	6.7	9.3	*11.4*	*17.1*	21.1
Android client	1359713	8.3	10.3	12.0	*17.3*	21.5
iPhone client	6222547	8.1	10.1	11.9	*17.6*	*21.6*
RIM client	3843404	7.6	10.0	11.8	*17.2*	21.1

Users' account history also illustrates several interesting trends. There is a clear trend towards stronger passwords amongst users who actively change their password, with users who have changed passwords 5 or more times being one of the strongest groups.[8] There is a weaker trend towards stronger passwords amongst users who have completed an email-based password recovery. However, users who have had their password reset manually after reporting their account compromised do not choose better passwords than average users.[9] Users who log in infrequently, judging by the last recorded login time before observation in our experiment, choose slightly better passwords. A much stronger trend is that users who have recently logged in from multiple locations choose relatively strong passwords.[10]

There is a weak trend towards improvement over time, with more recent accounts having slightly stronger passwords. Of particular interest to the security usability research community, however, a change in the default login form at Yahoo! appears to have had little effect. While Yahoo! has employed many slightly different login forms across different services, we can compare users who initially enrolled using two standard forms: one with no minimum length requirement or guidance on password selection, the other with a 6-character minimum and a graphical indicator of password strength. This change made no significant difference in security against online guessing and increased the offline metrics by only 1 bit.

Finally, we can observe variation between users who have actively used different Yahoo! services. Users who have used Yahoo!'s online retail platform (which means they have stored a payment card) do choose very weak passwords with lower frequency, with $\tilde{\lambda}_{10}$ increasing by about 2 bits. However, the distribution is indistinguishable from average users against offline attack. A similar phenomenon occurs for users of some other features, such as media sharing or dedicated smartphone clients for Android, RIM, or iOS, which achieve slightly better security against online attacks but are indistinguishable otherwise. Other popular features, such as webmail, chat, and social networking, saw slightly fewer weak passwords than normal, but again were indistinguishable against offline attacks.

One other interesting categorisation is the amount of data that users have stored with Yahoo!. While this is a very rough proxy for how active user accounts have been, there is a clear trend that users with a large amount of stored data choose better passwords.

6.3 Comparison with other password data sets

We now compare our collected data to several other leaked data sets (§D):

- RockYou (`www.rockyou.com`) is a social media and games website. It was compromised

[8] As these were voluntary changes, this trend does not support mandatory password change policies (§2.2.4).

[9] A tempting interpretation is that password choice does not play a significant role in the risk of account compromise, though this is not clearly supported since we can only observe the post-compromise strength.

[10] Yahoo! maintains a list of recent login locations for each user for abuse detection purposes.

data set	year	M	$\hat{\tilde{\lambda}}_1$	$\hat{\tilde{\lambda}}_{10}$	$\hat{\tilde{\lambda}}_{100}$	$\hat{\tilde{G}}_{0.25}$	$\hat{\tilde{G}}_{0.5}$
Yahoo	2011	69301337	6.5	9.1	11.4	17.6	21.6
RockYou	2009	32603388	6.8	8.9	11.1	15.9	19.8
Gawker	2010	748559	7.5	9.2	11.4	16.1	20.3
BHeroes	2011	548774	7.7	9.8	11.6	16.5	20.0

Table 6.2: Comparison of collected Yahoo data with leaked data sets.

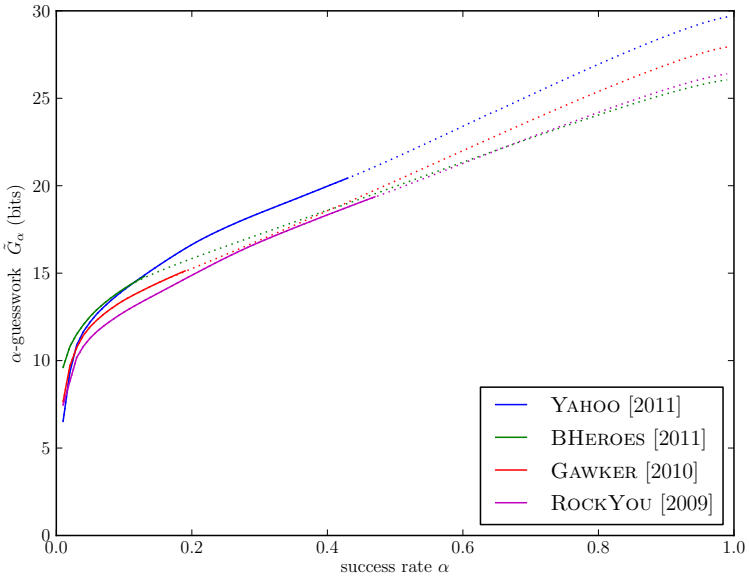
in December 2009 via SQL injection. Passwords were stored in plaintext and were leaked without associated identities. No password restrictions were in place.

- Gawker (`www.gawker.com`) is an online gossip blog. It was hacked in December 2010 and its complete database leaked. Passwords were hashed with traditional `crypt()` (§2.2.1), which incorporates 12 bits of salt. Thus, analysis of the distribution directly is not possible; the reported numbers are based on the best publicly available cracking efforts against the database. Also of note, `crypt()` limited all passwords to an effective length of 8 characters. No minimum length appears to have been in place.

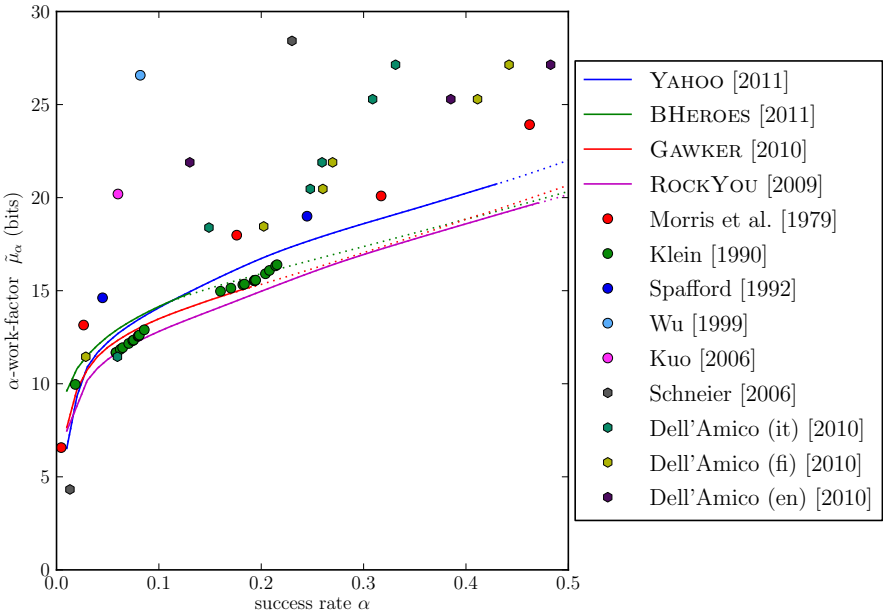
- BHeroes (`www.battlefieldheroes.com`) is a multiplayer online game. It was compromised in June 2011 by the LulzSec hacking group. Passwords were hashed using a single iteration of MD5 without any salt, making the distribution of hashes isomorphic to the distribution of underlying passwords. The site appears to have implemented a six-character minimum length for passwords; no passwords which have been cracked from the data set are less than six characters long.

Guessing metrics for all of these data sets as well as our aggregate Yahoo population are listed in Table 6.2. What is most striking is how consistent the estimated guessing metrics are. The distributions vary by less than 1 bit for $\hat{\tilde{\lambda}}_\beta$ for all $\beta \leq 100$, meaning they are very similar against online attack. Even against offline attack, extrapolated estimates for $\hat{\tilde{G}}_{0.5}$ vary by less than 2 bits. Some caution is in order as we have little evidence that the smaller distributions are well-approximated by our extrapolation method (§5.6) since the sample sizes are too small for cross-validation, though our model provides an equally good fit for subsamples of the RockYou data set. Guessing curves for all of these distributions are plotted in Figure 6.2a, demonstrating the relatively close estimates of $\tilde{G}$ for these data sets.

In Figure 6.2b we add data points from previous empirical estimates based on cracking (§2.5.1). In general, the cracking results produce higher estimated security than do statistical metrics. This is to be expected as the cracking tools used are not optimal. The 1990 cracking study by Klein provided estimates very close to the optimal attack for our observed data, suggesting either that this cracking approach was very well-tuned or that the underlying distribution was considerably weaker then.



(a) Full comparison of large password data sets



(b) Comparison of large password data sets to historical cracking results

Figure 6.2: Guessing curves for several large data sets of passwords. In Figure 6.2a, the guessing curve of $\tilde{G}_\alpha$ is plotted for the complete interval $0 \leq \alpha \leq 1$. The change to a dotted line indicates the point at which an extrapolated estimate is being used (§5.6). In Figure 6.2b, data points from historical cracking evaluations (§2.5.1) are added, necessitating a plot of $\tilde{\mu}_\alpha$ instead of $\tilde{G}_\alpha$.

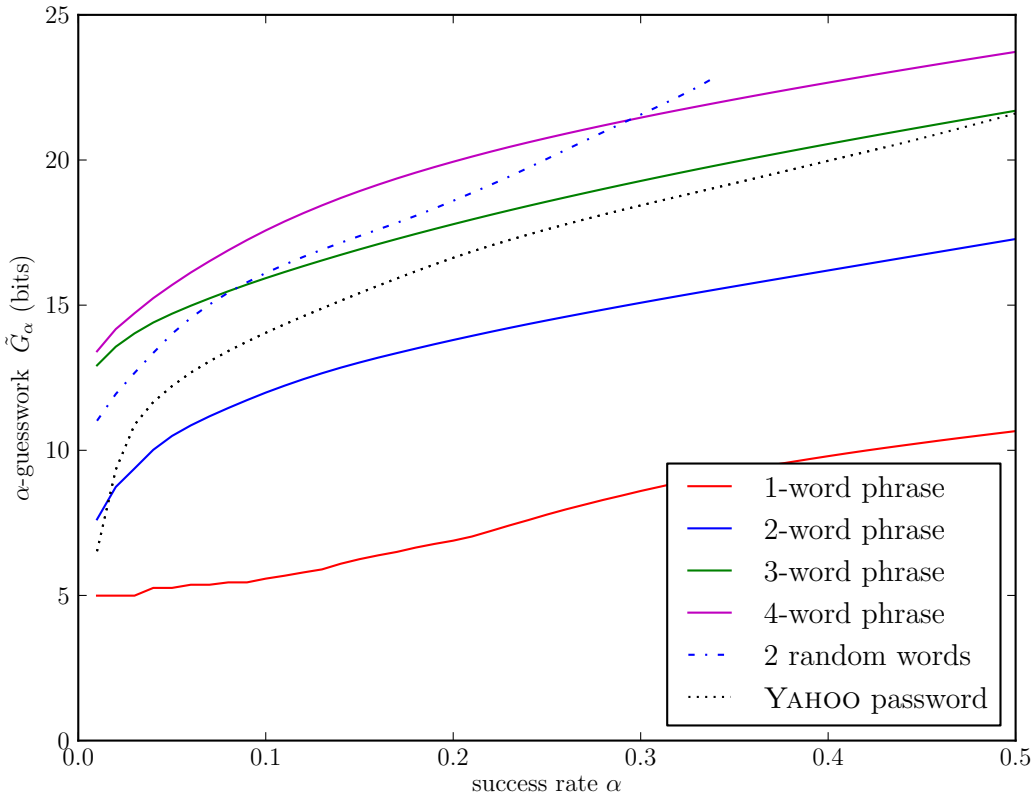


Figure 6.3: Comparison of Yahoo password data to sequences of English words.

6.4 Comparison with natural language patterns

A final question we can answer relatively easily is how password distributions compare to the distribution of words in natural language. This has long been of interest in password research, inspired by Shannon's 1951 experiment to calculate the entropy of individual characters in English text [272] which was used as a lower bound for the NIST password entropy formula [58].

We are interested in the distribution of English words and phrases. We can estimate this using data from the Google n-gram corpus which consists of over 10^{15} words of English-language text harvested from the World Wide Web in 2006 [52]. This corpus contains frequency counts for n-grams (sequences of n consecutive words) of up to 5 words. In Figure 6.3 we plot our password data sets against n-grams of 1–4 words. We also plot the head of the distribution of 2 words chosen individually according to the frequency of words in English.

Passwords appear to be between 2- and 3-word phrases in guessing difficulty, with a higher slope indicating that passwords are more skewed than English phrases (though drawn from a much larger space of possible values, since spelling and grammar aren't restrictions). Two English words chosen independently appear more difficult to guess than a text password.

Did you really name your son `Robert') DROP TABLE Students;--`*?*

—xkcd by Randall Munroe, 2007

Chapter 7

Guessing difficulty of personal knowledge questions

In this chapter we explore the difficulty of guessing answers to personal knowledge questions traditionally used for backup authentication. Most websites now prefer email-based password reset (§2.2.5), but personal knowledge questions are still used when email-based reset is not possible (for example, email accounts themselves) or as part of a multi-factor protocol [247].

An interesting aspect of personal knowledge questions is that there is often a *population-wide distribution* of answers which would be expected if all users answered accurately. For example, for the classic "What is your mother's maiden name?" question, a security engineer might expect the distribution of answers to be very similar to the population-wide distribution of surnames.[1] However, Rabkin found in a 2008 survey that 62% of users don't always answer personal knowledge questions accurately [243], either because a question wasn't applicable or because they wanted to provide a harder-to-guess answer.

This chapter will analyse data from large empirical distributions of answers to personal knowledge questions collected at Yahoo!, as well as several estimates for population-wide distributions to analyse the security impact of users not complying with intended question semantics.

[1] We'll discuss measuring the effect on an attacker using an approximation for the target distribution in §8. In this chapter we'll only focus on the difference in security caused by the user-chosen distribution differing from the population-wide distribution, assuming the attacker is optimal.

7.1 Sources of data

7.1.1 Questions of interest

It is difficult to quantitatively assess which questions are the most important in practice, which would require considering not only the varying popularity of different websites but which questions users prefer when given a choice. Two recent studies have surveyed questions with no data on user preference. Rabkin collected and classified 215 questions from 11 financial websites in 2008 [243].[2] Schechter et al. classified 29 questions from 4 large webmail providers (Microsoft, Google, AOL, and Yahoo!) in 2009 [261]. A third study, by Just and Aspinall in 2009 [163], allowed over 500 users to choose their own questions in a laboratory setting. We consider this data to be perhaps the most indicative of user preference, though all of these numbers are only meant as a rough guide for which questions are worthy of analysis.

While the variety of questions is large, many are asking for answers which have similar, well-defined population-wide distributions. For example, the population-wide distribution of answers to the question "What is your mother's maiden name" should be effectively equivalent to the question "What was your first school teacher's last name?" Thus we are most interested in classifying questions according to the distribution of answers they might induce and not the specific wording of the question. From the available empirical studies, we can divide questions into several important classes:

- *Surnames*, also called *last names* or *family names*. The distribution of surnames has been studied by historians and population geneticists [109] and statistics are well-known.
- *Forenames*, also called *first names* or *given names*. We also include middle names in this category; the distribution is similarly well-known.
- *Other names*, such as nicknames or terms of endearment.
- *Pet names*, sometimes specifically limited to dogs or cats.
- *Place names*, such as one's childhood address, hospital of birth, high school, or honeymoon destination. This is a much broader category than those above but population-wide distributions can be approximated by the populations of cities, schools, etc.
- *Favourites*, such as one's favourite sports team, hobby, television show, etc. It is much harder to find population-wide distributions for these questions.

Table 7.1 lists statistics estimated by the three empirical surveys for the frequency of these question types. None of the counts are complete because some questions don't fit into any of

[2] Of Rabkin's data set, roughly 50% (107 questions) came from a single website, Amtrust Direct. Thus it is far from a representative sample of banking practice.

category	Rabkin [243]	Schechter [261]	Just [163]
surname	4.7%	10.3%	$\top$
forename	21.8%	10.3%	34%
other name	4.7%	3.4%	$\bot$
pet name	3.3%	6.9%	15%
place	—	27.5%	20%
favourite	16.7%	41.4%	22%

Table 7.1: Common answer categories

these categories. We consider the estimates by Just and Aspinall to be the most accurate as they reflect what users actually choose; thus we consider names to be particularly important.

7.1.2 Yahoo! data

Our primary source of data are distributions collected from Yahoo! at the same time as the password experiment described in §6.1.5 was conducted. Unlike passwords, users' answers to personal knowledge questions are typically not stored hashed, but encrypted with a key available to customer service representatives who may use the plaintext answers to authenticate users over the telephone in order to help them to gain access to a lost account. Thus compiling the distribution of answers was relatively straightforward, unlike collecting password data which required a special experimental proxy to observe raw password data on submission.

A database crawl was executed which, for each (question, answer) pair, emitted the pair (question, $\mathbf{H}\,(\text{answer}||r)$), where r is a 128-bit random nonce used to ensure anonymity of the data as was used in the password collection experiment (§6.1). Questions chosen by fewer than 100,000 users were discarded. Including variants in different languages, this produced distributions of answers to 124 separate questions.

Separately, answers chosen by more than 1,000 users were emitted in plaintext. This value was chosen to allow limited analysis of semantic content while ensuring strong k-anonymity for any emitted data. All of the answers were stored in a canonical lower-case form with spacing and punctuation removed to prevent errors due to typos, but the full UTF-8 character set was allowed.

7.1.3 Population data

Population data was collected from government-collected census statistics where possible. A list of census sources is provided in §E. In addition, a large dataset of 269 million names was crawled from the online social network Facebook in 2010, which maintains a publicly-searchable (opt-out) list of registered users. This data set is perhaps the largest ever collected

for research and is more diverse than government statistics for any one country, though at the time of crawling just under half of Facebook's users were estimated to live in the USA.

7.2 Analysis of answers

We now analyse the empirical data from Yahoo!.There are many fewer hapax legomena observed in most distributions for personal knowledge questions than for passwords, meaning the interval of α for which we can compute guessing statistics with confidence using the methods introduced in §5 is much greater; for every distribution we studied we have confidence that the naive estimates for $\tilde{\mu}_\alpha$ and $\tilde{G}_\alpha$ are accurate to within 0.5 bits for all $\alpha \leq 0.5$.

7.2.1 Surnames

The distribution of surnames, listed in Table 7.2, varies significantly between countries, from South Korea where only three surnames (Kim, Park and Lee) are used by over half of the population to the USA, which has the greatest diversity at $\hat{\tilde{G}}_{0.5} = 12.0$ bits. Effects are not solely due to language, as there is significant variation between Chile and Spain, or between the USA, England and Australia. Ethnic diversity may be a major cause of these differences, with the USA as a nation of immigrants effectively having a mixture distribution of many different ethnic groups' surname distributions. Similarly, ethnic Russians make up over one-quarter of the population of Estonia which helps to explain Estonian surnames' relatively high guessing metrics (and the prominence of the surname Ivanov). The distribution of surnames on Facebook, which is likely to be even more diverse as a population of users from around the world, is the most difficult to guess by every measure.

Unlike for passwords, the shape of the guessing curves varies significantly for different distributions. For example, Finland's most common surname Virtanen is relatively uncommon compared to the most common surnames in other distributions, yet the distribution of Finnish surnames is much flatter and $\hat{\tilde{G}}_{0.5}$ is lower than other populations with individually more-common names.

Surnames were requested in two questions in the data collected from Yahoo!: "What is your mother's maiden name?" in English and "What is your father's second surname?" in Spanish.[3] Interestingly, the distribution of answers to the English question is weaker against guessing than either the US or Facebook population-wide distributions. Inspecting the top elements reveals `unknown`, `dontknow`, `notknown` and `none` as common answers.

The opposite effect occurred with Spanish-language second surnames, as the distribution of answers was stronger than population-wide distribution in Spain by about a bit and by an even

[3] In most Spanish-speaking countries people use two surnames, one from their father and one from their mother. Either parent's second surname can be used as a personal knowledge question, since neither is passed on to children. Women typically do not change names after marriage, preventing the maiden name formulation.

	x_1	$\lg M$	$\hat{\tilde{\lambda}}_1$	$\hat{\tilde{\lambda}}_{10}$	$\hat{\tilde{\lambda}}_{100}$	$\hat{\tilde{G}}_{0.25}$	$\hat{\tilde{G}}_{0.5}$
census records							
Australia	Smith	23.6	6.8	8.0	9.2	9.8	11.5
Chile	González	24.0	4.5	5.5	6.7	5.5	5.9
England	Smith	25.7	6.4	7.4	8.9	9.1	10.7
Estonia	Ivanov	20.4	7.6	8.4	9.7	10.8	11.5
Finland	Virtanen	22.3	7.8	8.1	9.0	9.2	10.3
Japan	Satō	24.8	6.0	6.7	8.3	7.7	9.0
Norway	Hansen	22.2	6.4	7.0	8.8	9.1	11.5
South Korea	Kim	25.5	2.2	3.9	6.6	2.4	3.0
Spain	Garcia	25.5	5.0	5.8	8.0	6.3	8.5
USA	Smith	28.1	6.9	7.7	9.3	10.0	12.0
crawled							
Facebook	Smith	28.0	8.0	8.6	9.9	11.5	13.7
Yahoo! personal knowledge question answers							
mother's maiden (en)	`unknown`	20.2	6.2	7.3	8.9	9.3	11.2
father's second surname (es)	`garcia`	18.2	5.9	6.5	8.3	7.6	9.5

Table 7.2: Guessing metrics for distributions of surnames.

larger margin compared to Chile. This may be partially a diversity effect, as Spanish-speakers from many countries use Yahoo!. Another difference is that the most common non-compliant response `notiene` ("he doesn't have one") was the 56^{th} most common, compared to `unknown` which was the most common response in English for mother's maiden name. Finally many users drop stress accents when recording their answer. For example, the most common response `garcia` also appeared in the proper form `garcía` (though only one-sixth as often). Variation in entry can only increase guessing difficulty.

7.2.2 Forenames

Many of the observations for population-wide distributions of surnames carry over to forenames, as listed in Table 7.3, though fewer national governments publish statistics on forename distributions so fewer countries are included. There are again differences between different countries, though we could not identify an outlier as significant as South Korea was for surnames. Ethnic diversity may again play a role with the USA and Belgium (which contains both large Dutch- and French-speaking populations) having more variation than Spain, and the distribution found on Facebook again being the hardest to guess. It is also possible to explore the difference between male and female naming patterns; for all three countries examined there is more variation in female names than male names.

	x_1	$\lg M$	$\hat{\tilde{\lambda}}_1$	$\hat{\tilde{\lambda}}_{10}$	$\hat{\tilde{\lambda}}_{100}$	$\hat{\tilde{G}}_{0.25}$	$\hat{\tilde{G}}_{0.5}$
census records (overall)							
Belgium	Maria	23.2	5.7	6.9	8.2	7.8	8.5
Spain	María	25.5	3.5	5.3	7.5	5.2	7.0
USA	Michael	27.8	5.9	6.6	8.0	7.5	8.4
crawled (overall)							
Facebook	David	28.0	7.5	8.0	9.1	9.3	10.5
census records (female)							
Belgium	Maria	22.3	4.9	6.4	7.9	7.2	8.1
Spain	María	24.5	2.5	5.0	7.3	4.4	6.3
USA	Jennifer	26.7	6.3	6.8	7.9	7.3	8.2
census records (male)							
Belgium	Jean	22.3	5.7	6.5	7.7	6.9	7.6
Spain	José	24.4	3.4	4.9	7.3	4.4	5.8
USA	Michael	26.8	5.0	5.7	7.4	5.9	6.7
Yahoo! personal knowledge question answers							
father's name (es)	`jose`	17.5	5.1	5.8	7.7	6.2	7.5
father's name (it)	`giuseppe`	17.1	4.8	5.5	7.4	5.6	6.7
father's middle name (en)	`edward`	22.7	6.0	6.5	8.2	7.4	9.7
father's middle name (pt)	`antonio`	17.1	5.6	6.3	8.0	7.0	8.6
your middle name (fr)	`marie`	17.8	4.5	6.4	8.1	7.5	8.6

Table 7.3: Guessing metrics for distributions of forenames.

All of the empirical distributions of responses to questions requesting a forename appear slightly more difficult-to-guess than population-wide baselines might suggest (though lower than the overall distribution of forenames on Facebook.) There is no evidence for any explanation except strong non-compliant answers. Amongst the answers for which plaintext was available, only `none` as the sixth most common response in English for one's father's middle name and `aucun` (none) as the eighth most common response in French for one's own middle name appeared to be weak defaults. All other common answers appeared legitimate.

7.2.3 Pet names

Guessing metrics for names of pets are listed in Table 7.4. This question was asked with a generic "what is your pet's name" phrasing in many different languages. The only population-wide statistics available are those kept by city pet-registration departments, of which we found three, all in the USA.

The population-wide distribution of pet names has higher guessing metrics than do human

	x_1	$\lg M$	$\hat{\tilde{\lambda}}_1$	$\hat{\tilde{\lambda}}_{10}$	$\hat{\tilde{\lambda}}_{100}$	$\hat{\tilde{G}}_{0.25}$	$\hat{\tilde{G}}_{0.5}$
city registration records							
Des Moines	buddy	14.9	6.2	7.0	8.4	8.0	9.4
Los Angeles	lucky	19.0	6.4	6.9	8.3	7.8	9.2
San Francisco	buddy	15.7	6.7	7.2	8.5	8.2	9.5
Yahoo! personal knowledge question answers							
Chinese	`mimi`	19.9	7.1	7.9	9.1	9.7	12.6
English	`lucky`	24.3	7.5	8.0	9.2	9.7	11.5
French	`chat`	17.1	4.0	6.1	8.4	7.8	9.9
German	`hund`	17.6	6.0	7.0	8.5	8.3	9.8
Italian	`cane`	17.5	5.4	6.8	8.5	8.2	9.8
Portuguese	`cachorro`	19.1	3.8	6.1	8.4	7.8	9.7
Spanish	`perro`	20.1	6.2	7.4	9.0	9.3	10.7

Table 7.4: Guessing metrics for distributions of pet names.

forenames. Similarly, empirical responses appear slightly better than forenames when measured by $\hat{\tilde{G}}_{0.25}$ or $\hat{\tilde{G}}_{0.5}$, but much lower when measured by $\hat{\tilde{\lambda}}_1$ due to a large number of users simply stating the species of their pet—the words for "dog" and "cat" were the most popular responses in French, German, Italian, Portuguese, and Spanish. In Portuguese over 7% of users simply answered `cachorro` ("dog"). This pattern did not hold in English, where `dog` and `cat` were the 23$^{\text{rd}}$ and 51$^{\text{st}}$ most common responses, behind `ahmed` at 20$^{\text{th}}$. Finally, the distribution of Chinese responses is even more difficult to guess. It contains a mixture of responses in English, Pinyin,[4] and Chinese characters, as well as many non-compliant responses such as `123456`, which was the tenth most common response.

7.2.4 Place names

Table 7.5 lists guessing metrics for several questions requesting place names. These questions don't have nearly as well-defined of a population-wide distribution as do human names, but the empirical answers appear generally harder to guess than do names. In particular, the name of a person's first school is amongst the strongest of any questions asked, consistent across languages. School names may gain resistance to guessing from the fact there are limits as to how large primary schools can grow (though some schools do share common names), unlike cities whose populations are conjectured to follow a power-law distribution [69].

[4]Pinyin is the standard system to transliterate Chinese characters into the Latin alphabet. For example, the most common response `mimi` for a pet name is the Pinyin transliteration for "meow."

	x_1	$\lg M$	$\hat{\tilde{\lambda}}_1$	$\hat{\tilde{\lambda}}_{10}$	$\hat{\tilde{\lambda}}_{100}$	$\hat{\tilde{G}}_{0.25}$	$\hat{\tilde{G}}_{0.5}$
first school							
Chinese	—	19.3	7.5	8.5	10.0	11.5	13.6
English	`stmarys`	22.6	8.9	9.5	10.8	13.2	15.3
Portuguese	`sesi`	17.4	6.6	8.5	9.9	11.4	13.3
Spanish	`lasalle`	18.3	7.5	8.4	9.7	11.0	12.9
place of meeting partner/spouse							
Chinese	—	17.9	4.4	6.4	8.6	8.5	13.4
English	`school`	21.6	4.8	6.2	8.6	8.3	11.7
Spanish	`trabajo`	18.3	5.6	6.4	8.4	7.7	10.7
other places (all English)							
city of birth	`chicago`	20.2	6.7	7.4	8.7	8.7	10.3
childhood summer home	`home`	18.6	4.4	6.3	8.5	8.0	10.7
childhood street	`main`	19.8	9.1	9.8	11.0	12.3	13.8
hospital of birth	`home`	19.5	7.0	7.8	9.6	11.0	11.8

Table 7.5: Guessing metrics for distributions of places.

7.2.5 Favourites

Table 7.6 lists guessing metrics for questions requesting a user's favourite item. Again, there is no natural source for a population-wide distribution for these items. Several are surprisingly strong by $\hat{\tilde{G}}_{0.5}$, for example the distribution of favourite restaurants is better than most distributions of surnames. All of the favourite questions have several very common answers though, with one's favourite colour or sports team being the weakest distributions observed.

7.3 Security implications

None of the personal knowledge questions examined in this chapter provides useful resistance to offline guessing. The security provided is roughly comparable to PINs against online attacks, equivalent to a 5–10 bit uniform distribution for most questions, and generally much weaker against offline attack than passwords, equivalent to less than 15 bits by $\hat{\tilde{G}}_{0.5}$. Unlike for passwords, in our large Facebook distributions for forenames and surnames we are confident in our partial guessing metrics for $\alpha = 0.9$ without relying on any assumptions about the distribution of names.

Still, personal knowledge questions may be difficult enough to guess to play a role in a broader account recovery process given firm rate-limiting. Unlike with passwords there is large variation in distributions between different language speakers and even larger variation between

	x_1	$\lg M$	$\hat{\tilde{\lambda}}_1$	$\hat{\tilde{\lambda}}_{10}$	$\hat{\tilde{\lambda}}_{100}$	$\hat{\tilde{G}}_{0.25}$	$\hat{\tilde{G}}_{0.5}$
favourite hero							
Chinese	`leifeng`	17.0	5.5	6.5	8.5	8.1	12.0
English	`superman`	21.6	3.7	5.7	8.2	6.6	10.2
French	`jesus`	16.9	4.8	6.2	8.3	7.5	10.2
Portuguese	`meupai`	16.7	4.1	5.5	7.9	5.8	8.6
Spanish	`superman`	17.3	3.3	5.3	7.8	5.3	8.2
favourite hobby							
English	`reading`	21.4	4.2	5.6	7.8	5.8	8.0
French	`football`	16.7	3.9	5.4	7.6	5.4	7.3
Portuguese	`musica`	17.6	5.3	5.8	7.8	6.3	8.1
Spanish	`leer`	17.9	3.6	5.2	7.6	4.9	7.0
favourite sports team							
English	`india`	21.4	4.8	5.8	7.5	6.0	7.0
Portuguese	`flamengo`	19.3	3.0	4.3	7.1	3.7	4.2
Spanish	`realmadrid`	18.1	4.0	4.7	7.1	4.2	5.2
favourite sport							
Chinese	(basketball)	16.9	4.5	5.6	7.8	5.7	8.0
French	`football`	17.5	1.7	4.0	6.9	1.7	2.8
other favourite items (all English)							
author	`eminescu`	19.0	6.3	7.2	9.0	9.7	10.9
book	`bible`	19.2	4.4	5.7	8.4	7.2	10.4
car	`honda`	20.8	4.3	5.4	7.7	5.5	7.4
colour	`blue`	17.2	1.3	3.5	6.7	1.3	1.8
restaurant	`mcdonalds`	17.6	5.7	6.6	8.8	9.1	12.3
musician	`arrahman`	20.3	6.9	7.4	8.9	9.2	11.2

Table 7.6: Guessing metrics for distributions of favourite items.

different questions, meaning that careful study is necessary to pick the best questions possible. The most promising approach may be to avoid human-chosen distributions of names and instead ask for items like primary schools for which there are fewer very-popular items.

You go to war with the army you have—
not the army you might want or wish to have at a later time.

—Donald Rumsfeld, 2004

Chapter 8

Sub-optimal guessing attacks

In this chapter, we consider a sub-optimal guessing attack against values drawn from $\mathcal{X}$ based on an assumed distribution $\mathcal{Y} \neq \mathcal{X}$. In practice, an attacker is unlikely to have perfect knowledge of $\mathcal{X}$, as we assumed when calculating guessing metrics in §4 and §§6–7. We introduce simple extensions to our guessing metrics to capture the expected loss of efficiency for a guessing attack based on an incorrect dictionary and compute values for data sets seen so far.

8.1 Divergence metrics

We'll introduce *divergence metrics* for an attack against $\mathcal{X}$ (unknown to the attacker) using $\mathcal{Y}$ as an assumed distribution; we'll denote this scenario as $\mathcal{X} \parallel \mathcal{Y}$. We'll write $p_j^{\mathcal{X}}$ for the probability in $\mathcal{X}$ of the j^{th} most common event in $\mathcal{Y}$, and $p_j^{\mathcal{Y}}$ for the probability in $\mathcal{Y}$ of the same event.

Note that we are now using subscript j (instead of i) to indicate that it always refers to the index from the distribution $\mathcal{Y}$. It is critical that $p_j^{\mathcal{X}}$ and $p_j^{\mathcal{Y}}$ refer to different probabilities for the same event; we choose the indices from $\mathcal{Y}$ to retain the convention that values are guessed by the attacker in order of increasing j.

8.1.1 Kullback-Leibler divergence and cross entropy

A related problem first studied by Solomon Kullback and Richard Leibler in 1951 [180] is the loss of efficiency when compressing data using a code designed for a different distribution of values. Recalling that the Shannon entropy $H_1(\mathcal{X})$ indicates the expected number of bits needed to encode an event drawn from $\mathcal{X}$ using an optimal code (§3.1.1), the *cross entropy* of $\mathcal{X} \parallel \mathcal{Y}$ measures the expected number of bits needed to encode an event drawn from $\mathcal{X}$ using

an optimal coding scheme based on $\mathcal{Y}$:

$$H_1(\mathcal{X} \parallel \mathcal{Y}) = \sum_{j=1}^{|\mathcal{Y}|} p_j^{\mathcal{X}} \lg p_j^{\mathcal{Y}} \tag{8.1}$$

This equation is very similar to the basic definition of Shannon Entropy (Equation 3.1). Because the optimal coding scheme will encode the j^{th} event (in $\mathcal{Y}$) using $\lg p_j^{\mathcal{Y}}$ bits, this is simply an expectation of the number of bits to encode an event drawn from $\mathcal{X}$.

An alternative metric is the number of *extra bits* needed to encode events from $\mathcal{X}$ using a code from $\mathcal{Y}$. This value is called the *Kullback-Leibler divergence*:

$$D_{\text{KL}}(\mathcal{X} \parallel \mathcal{Y}) = \sum_{j=1}^{|\mathcal{Y}|} p_j^{\mathcal{X}} \lg \frac{p_j^{\mathcal{Y}}}{p_j^{\mathcal{X}}} \tag{8.2}$$

When $\mathcal{X} = \mathcal{Y}$, we will trivially have $H_1(\mathcal{X} \parallel \mathcal{Y}) = H_1(\mathcal{X})$ and $D_{\text{KL}}(\mathcal{X} \parallel \mathcal{Y}) = 0$. In general:

$$D_{\text{KL}}(\mathcal{X} \parallel \mathcal{Y}) = H_1(\mathcal{X} \parallel \mathcal{Y}) - H_1(\mathcal{X}) \tag{8.3}$$

It is also important to note that both $H_1(\mathcal{X} \parallel \mathcal{Y})$ and $D_{\text{KL}}(\mathcal{X} \parallel \mathcal{Y})$ are undefined in our notation if there are events in $\mathcal{X}$ which don't exist in $\mathcal{Y}$. This occurs because if $p_j^{\mathcal{Y}}$ is 0 for some $j \leq |\mathcal{Y}|$, then both values will be ∞ because they include $\lg 0$.

This appropriately reflects that it is impossible to encode an event which wasn't incorporated into the design of an optimal code. This is another reason why H_1 is inappropriate as a guessing attacks as it indicates that guessing is infinitely difficult without perfect knowledge of the underlying distribution. Similarly, by the bound in Equation 3.17, G_1 will indicate that such a guessing attack is infinitely difficult.

8.1.2 Divergence for partial guessing metrics

It is straightforward to define cross variants for all of the partial guessing metrics introduced in §3.2. We begin with β-success-rate (§3.2.1), which is the simplest:

$$\lambda_\beta(\mathcal{X} \parallel \mathcal{Y}) = \sum_{j=1}^{\beta} p_j^{\mathcal{X}} \tag{8.4}$$

Similarly for α-work-factor (§3.2.2):

$$\mu_\alpha(\mathcal{X} \parallel \mathcal{Y}) = \min\left\{ \mu \middle| \sum_{j=1}^{\mu} p_j^{\mathcal{X}} \geq \alpha \right\} \tag{8.5}$$

Recall that for α-guesswork (§3.2.3), the definition includes the rounded-up $\lceil\!\lceil \alpha \rceil\!\rceil$ (Equation 3.11). In defining a cross version of G_α, we'll need to define $\lceil\!\lceil \alpha \rceil\!\rceil$ in terms of $\mathcal{Y}$:

$$\lceil\!\lceil \alpha \rceil\!\rceil = \lambda_{\mu_\alpha(\mathcal{X} \parallel \mathcal{Y})}(\mathcal{X} \parallel \mathcal{Y}) = \sum_{j=1}^{\mu_\alpha(\mathcal{X})} p_j^{\mathcal{X}} \tag{8.6}$$

Using this, the cross α-guesswork is very similar to the original definition (Equation 3.10):

$$G_\alpha\left(\mathcal{X} \parallel \mathcal{Y}\right) = \left(1 - \lceil\!\lceil \alpha \rceil\!\rceil\right) \cdot \mu_\alpha\left(\mathcal{X} \parallel \mathcal{Y}\right) + \sum_{j=1}^{\mu_\alpha(\mathcal{X}\|\mathcal{Y})} p_j^{\mathcal{X}} \cdot j \tag{8.7}$$

Bit-converted metrics $\tilde{\lambda}_\beta\left(\mathcal{X} \parallel \mathcal{Y}\right)$, $\tilde{\mu}_\alpha\left(\mathcal{X} \parallel \mathcal{Y}\right)$ and $\tilde{G}_\alpha\left(\mathcal{X} \parallel \mathcal{Y}\right)$ can be computed using the definitions in §3.2.4 exactly, using Equation 8.6 for $\lceil\!\lceil \alpha \rceil\!\rceil$ where needed.

Defining divergence concisely like in Equation 8.2 for D_{KL} is difficult for partial metrics because the logarithm in the bit-conversion formulas is taken after a summation. It is simpler to use the equivalence that:

$$D_{\mathcal{P}}\left(\mathcal{X} \parallel \mathcal{Y}\right) = \mathcal{P}\left(\mathcal{X} \parallel \mathcal{Y}\right) - \mathcal{P}(\mathcal{X}) \tag{8.8}$$

where $\mathcal{P}$ is any of the properties we have introduced.

8.1.3 Sample size

The effects of sample size are similar to those we explored in §5 for optimal guessing metrics. In Figure 8.1 we plot the cross α-guesswork for the full YAHOO data set, using a random subsample $\text{YAHOO}_{1\text{M}}$ with one million samples as both the assumed distribution and target distribution. Sample size makes no significant impact for $\alpha < \alpha_*$, the interval for which we are confident in our estimates of $\tilde{\mu}_\alpha$ (§5.5).

For $\alpha > \alpha_*$, observe that $\hat{\tilde{\mu}}_\alpha\left(\text{YAHOO}_{1\text{M}} \parallel \text{YAHOO}_{69\text{M}}\right) \approx \hat{\tilde{\mu}}_\alpha\left(\text{YAHOO}_{69\text{M}}\right)$. Using the larger, more-accurate data set to conduct a guessing attack against the less-accurate one causes no loss of efficiency compared to attacking the complete distribution with perfect knowledge.

However, conducting an attack on the full data set using the smaller data set does cause a loss of efficiency, that is $\hat{\tilde{\mu}}_\alpha\left(\text{YAHOO}_{69\text{M}} \parallel \text{YAHOO}_{1\text{M}}\right) > \hat{\tilde{\mu}}_\alpha\left(\text{YAHOO}_{69\text{M}}\right)$ for some $\alpha < \alpha_*$ Specifically, the sampled data set will be missing many items which occurred more than once in the full data sets, forcing less-probable values to be guessed earlier. Furthermore, we can't estimate $\hat{\tilde{\mu}}_\alpha\left(\text{YAHOO}_{69\text{M}} \parallel \text{YAHOO}_{1\text{M}}\right)$ at all for higher values of α because our model doesn't allow the attacker to guess any values not seen in $\text{YAHOO}_{1\text{M}}$.

When analysing a sub-optimal attack for two sampled distributions $\hat{\mathcal{X}} \parallel \hat{\mathcal{Y}}$, we are thus concerned mainly with the inaccuracy in $\hat{\mathcal{Y}}$ and not $\hat{\mathcal{X}}$. In the next section we'll take a conservative approach and only take estimates for events with frequency $f \geq m^*$, the calculated minimum frequency for which we expect our estimates of $\hat{\tilde{\mu}}_\alpha$ to be biased by less than 0.5 bits (§5.5).

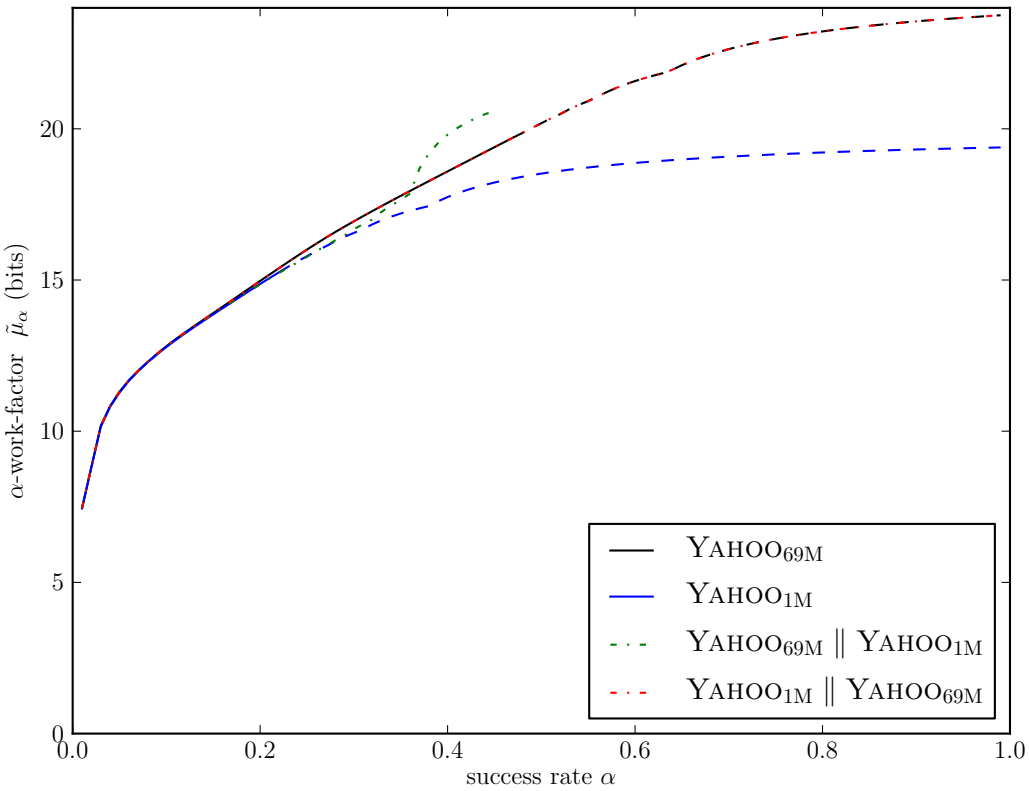


Figure 8.1: Effect of sample size on estimating the cross α-work-factor using the original YAHOO data set (69 M samples) and subsamples taken with 1M samples.

8.2 Applications

8.2.1 Sub-optimal attacks on 4-digit PINs

As a first example, the cross α-guesswork is plotted in Figure 8.2 for the ROCKYOU-4 and IPHONE distributions of numeric PINs introduced in §4.1. Both distributions had `1234` as the most common element, so there is no loss of efficiency for attacks with only $\beta = 1$ guesses. The next few items vary greatly though between the two distributions: `0000` is the second most-common PIN in the IPHONE set with $p_{0000}^{\text{IPHONE}} = 2.56\%$ but only $p_{0000}^{\text{ROCKYOU}} = 0.11\%$, while the second most-common PIN in the ROCKYOU set is `2007` with $p_{2007}^{\text{ROCKYOU}} = 2.22\%$ but only $p_{2007}^{\text{IPHONE}} = 0.04\%$. Using either distribution to guess values against the other there is an increase in difficulty of $D_{\tilde{\lambda}_\alpha} > 3$ bits for $\alpha = 0.1$, roughly a decimal order of magnitude loss of efficiency. Thus, training on the wrong data set can make a significant difference in efficiency for the most important type of guessing attack against PINs. It is also evident in Figure 8.2 that the cross α-guesswork is no longer a monotonically increasing function with α as the attacker is no longer guessing in optimal order.

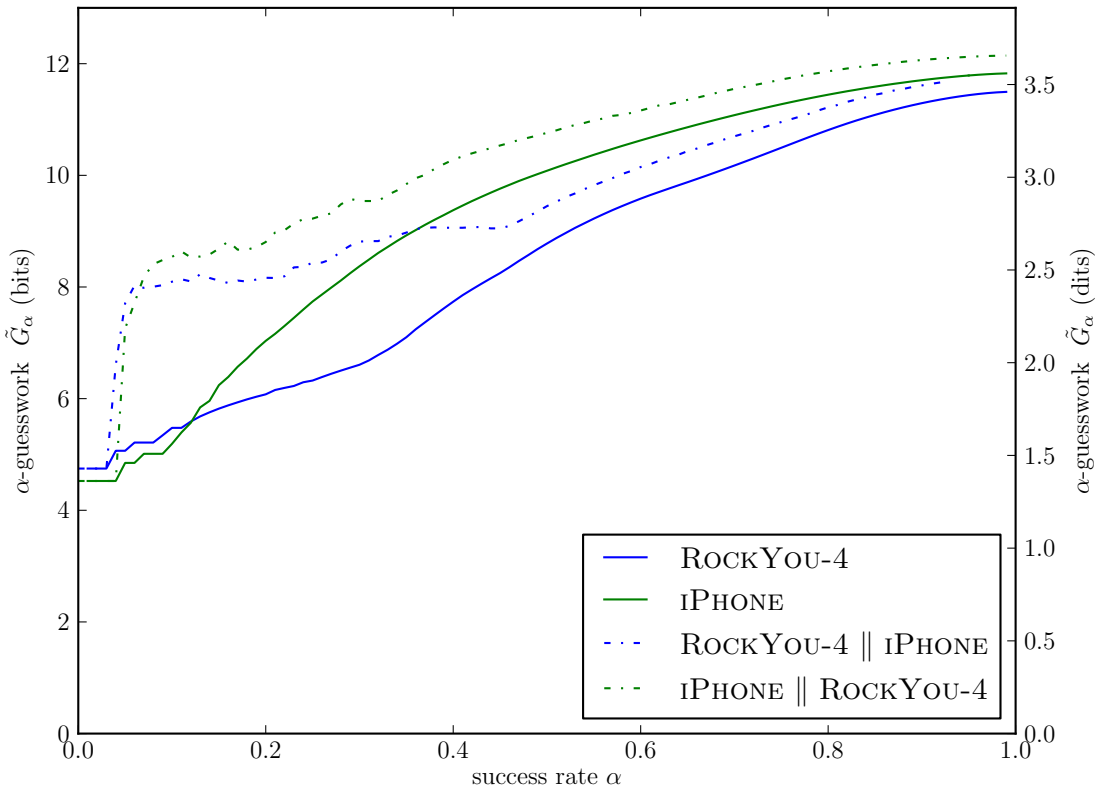


Figure 8.2: Effect of a sub-optimal dictionary when guessing PINs. The cross α-guesswork is plotted for both the RockYou-4 and iPhone data sets using the other as an approximate distribution.

8.2.2 Demographic differences at Yahoo!

Returning to the Yahoo data set of passwords, we are interested in the efficiency of an attack trained on the wrong demographic subset of Yahoo! users. For consistency in this section, we will consider $\lambda_{1000}(\mathcal{X} \parallel \mathcal{Y})$ which is within the well-approximated region for all of our demographic subsets. A simple example is male and female-chosen passwords:

		dictionary	
		♀	♂
target	♀	**7.8%**	6.8%
	♂	6.3%	**7.1%**

There is a 10–15% loss in efficiency if an attacker uses the optimal male dictionary against female-chosen passwords, or vice-versa, a divergence of only 0.1 bits. This is small enough that we may conclude real-world attackers are unlikely to tailor their guessing approach based on the gender distribution of their target users.

target \ dictionary	Chinese	German	Greek	English	French	Indonesian	Italian	Korean	Portuguese	Spanish	Vietnamese	global	minimax
Chinese	**4.4%**	1.9%	2.7%	2.4%	1.7%	2.0%	2.0%	2.9%	1.8%	1.7%	2.0%	2.9%	2.7%
German	2.0%	**6.5%**	2.1%	3.3%	2.9%	2.2%	2.8%	1.6%	2.1%	2.6%	1.6%	3.5%	3.4%
Greek	9.3%	7.7%	**13.4%**	8.4%	7.4%	8.1%	8.0%	8.0%	7.7%	7.8%	7.7%	8.6%	8.9%
English	4.4%	4.6%	3.9%	**8.0%**	4.3%	4.5%	4.3%	3.4%	3.5%	4.2%	3.5%	7.9%	7.7%
French	2.7%	4.0%	2.9%	4.2%	**10.0%**	2.9%	3.2%	2.2%	3.1%	3.4%	2.1%	5.0%	4.9%
Indonesian	6.7%	6.3%	6.5%	8.7%	6.3%	**14.9%**	6.2%	5.8%	6.0%	6.2%	5.9%	9.3%	9.6%
Italian	4.0%	6.0%	4.6%	6.3%	5.3%	4.6%	**14.6%**	3.3%	5.7%	6.8%	3.2%	7.2%	7.1%
Korean	3.7%	2.0%	3.0%	2.6%	1.8%	2.3%	2.0%	**5.8%**	2.4%	1.9%	2.2%	2.8%	3.0%
Portuguese	3.9%	3.9%	4.0%	4.3%	3.8%	3.9%	4.4%	3.5%	**11.1%**	5.8%	2.9%	5.1%	5.3%
Spanish	3.6%	5.0%	4.0%	5.6%	4.6%	4.1%	6.1%	3.1%	6.3%	**12.1%**	2.9%	6.9%	7.0%
Vietnamese	7.0%	5.7%	6.2%	7.7%	5.8%	6.3%	5.7%	6.0%	5.8%	5.5%	**14.3%**	7.8%	8.3%

Table 8.1: Language dependency of password guessing. Each cell indicates the success rate $\lambda_{1000}(\text{target} \parallel \text{dictionary})$ of a guessing attack with 1000 attempts. The greatest efficiency loss, when attacking French passwords using a Vietnamese dictionary, is only a factor of 4.8, or $D_{\tilde{\lambda}_{1000}} = 2.26$ bits.

Different language communities would seem to be the most likely to diverge significantly, though in Table 8.1 we see surprisingly little loss of efficiency for an attack trained on passwords chosen by speakers of a different language. The worst divergence observed is only 2.26 bits, when using an optimal Vietnamese-language password dictionary against French speakers' passwords.

We also observe in Table 8.1 that simply using the global list of most popular passwords performs very well against most subsets. The greatest divergence when using the global list is only 1.12 bits, against Portuguese-language passwords. We can improve this slightly further by constructing a special dictionary to be effective against all subsets. We do this by repeatedly choosing the password for which the lowest popularity in any subset is maximal and call it the *minimax dictionary*, also seen in Table 8.1. This dictionary performs very similarly to the global dictionary, though is slightly worse for some subsets but better for those which the global list is worst for. The worst-case divergence for the minimax dictionary is just 1.06 bits, also for Portuguese-language passwords.

Digging into our data, we can identify some "global passwords" which are popular across all subgroups. The single most popular password we observed, for example, occurred with probability at least 0.14% in every sub-population. Some overall popular passwords were very rare in certain sub-populations. For example, the third most common password, with overall probability 0.1%, occurred nearly 100 times less frequently in some sub-populations. However, there were eight passwords which occurred with probability at least 0.01% in every sub-population. Without access to the raw passwords, we can only speculate that these are numeric passwords as these are popular and internationalise well.[1]

Despite the existence of globally popular passwords, however, we still consistently find small divergences between seemingly similar lists of passwords. For example, efficiency losses of up to 25% can occur using dictionaries tailored to people from different English-speaking countries:

		dictionary				
		us	uk	ca	au	global
target	us	**8.2%**	6.6%	7.4%	7.2%	8.1%
	uk	5.4%	**6.9%**	5.5%	5.6%	5.5%
	ca	8.8%	7.9%	**9.9%**	8.7%	8.8%
	au	7.4%	7.2%	7.6%	**8.8%**	7.5%

[1] Within the RockYou data set, `123456` is the most popular password and 5 other numeric-only passwords are amongst the top ten.

We can observe similar efficiency losses based on age:

		dictionary				
		13–20	21–34	35–54	55+	global
target	13–20	**8.4%**	7.8%	7.1%	6.5%	7.9%
	21–34	7.3%	**7.9%**	7.3%	6.7%	7.8%
	35–54	5.4%	5.8%	**6.4%**	6.1%	6.2%
	55+	5.4%	5.8%	6.8%	**7.3%**	6.5%

Finally, we observe efficiency losses up to 25% based on service usage:

		dictionary				
		retail	chat	media	mail	global
target	retail	**7.0%**	5.6%	6.6%	5.6%	6.0%
	chat	6.9%	**8.4%**	7.8%	8.3%	8.3%
	media	5.7%	5.6%	**6.0%**	5.6%	5.8%
	mail	6.7%	8.0%	7.5%	**8.2%**	8.1%

8.2.3 Real password guessing attacks

Ideally, we could measure the efficiency of realistic guessing attacks against a large data set like RockYou.[2] It is difficult to do so in a canonical way though because password cracking tools are highly tunable. Furthermore, most have now been modified based on the RockYou data set and other large leaks, spoiling the experiment.

We use several independent proxies for guessing attacks instead. Two studies, by Seifert [269] and Owens [229] have collected failed login attempts on SSH servers and reported the most likely passwords attempted in guessing attacks. We can also extract password lists from prominent samples of malware, for example the Morris worm [278] or Conficker [284], both of which contain small dictionaries of passwords. Finally we can consider the list of passwords banned by the microblogging website Twitter, which checks passwords in client-side JavaScript in an unusual arrangement that makes it possible to study the exact blacklist.

We report on the efficiency of an attack against the RockYou passwords using all of these data sets in Table 8.2. We also include, as a baseline for comparison, the 1000 most popular passwords in the BHeroes data set and the 1000 most popular passwords from the 70yx data set, which was leaked from a Chinese-language website (§D).

[2]We can't use the Yahoo data set for this experiment since, by design, there is no way to test for the existence of specific passwords.

year	list	M	$\tilde{\lambda}_1$	$\min\left(\tilde{\lambda}_1\right)$	$\tilde{\lambda}_{10}$	$\min\left(\tilde{\lambda}_{10}\right)$	$D_{\tilde{\lambda}_M}$
			SSH guessing attacks				
2006	Seifert [269] (campus)	20	6.81	6.81	9.46	9.33	0.68
2006	Seifert [269] (business)	20	6.81	6.81	9.86	9.51	0.86
2006	Seifert [269] (residence)	20	6.81	6.81	9.56	9.45	0.81
2008	Owens [229]	28	6.81	6.81	10.06	9.72	1.24
			passwords dictionaries in malware				
1989	Morris worm [278]	431	—	9.10	—	11.00	2.66
2009	Conficker [284]	181	—	6.81	—	9.07	1.21
			password blacklists				
2010	Twitter	396	—	6.81	—	9.43	2.37
2011	Twitter	396	—	6.81	—	9.08	0.58
			baselines				
2011	BHeroes	1000	6.81	6.81	9.40	9.13	0.74
2012	70yx	1000	6.81	6.81	9.96	9.27	2.36
			optimal				
2010	RockYou	1000	6.81	6.81	8.93	8.93	0

Table 8.2: Efficiency for several proxies for real guessing attacks. Because many of these lists are not sorted, we assume an optimal ordering of the dictionary.

Unfortunately, the blacklists and the lists in malware are unordered. Thus, we can only compute $\tilde{\lambda}_\beta$ for an attack using these lists assuming the attacker uses an optimal order, denoted $\min\left(\tilde{\lambda}_\beta\right)$. Table 8.2 lists values for $\beta = 1, 10$ and as well as $D_{\tilde{\lambda}_M}$, the divergence for an attack using the whole list. For an attacker using the whole list, the order of guesses won't affect $\tilde{\lambda}_\beta$.

The password `123456` was included in every set except that carried by the 1989 Morris worm (which included no numeric passwords) and was listed first in every ordered set, giving the optimal value of $\tilde{\lambda}_1 = 6.81$ bits. Efficiency for 10 guesses, measured by $\tilde{\lambda}_{10}$, was never more than a bit higher than optimal. Thus, it appears that very limited online guessing attacks are well-understood and near-optimally executed.

For a few of the larger lists, the divergence from optimal became greater when the whole list was taken into account. For example, Twitter's 2010 blacklist was 2.37 bits away from ideal, more than a factor of 5, and included several bizarre passwords not seen in the RockYou data set at all. The 2011 update was a considerable improvement. The Morris worm's list is inadequate because it only included lower-case letters in its passwords, whereas numeric passwords represent many of the most common. Finally, the top 1000 passwords in the 70yx diverged by about 2.36 bits for $\beta = 1000$, similar to the worst-case numbers in Table 8.1 for the gap between passwords chosen in different languages.

Always remember that you are absolutely unique—
just like everybody else.

—Margaret Mead, 1970s (apocryphal)

Chapter 9

Individual-item strength metrics

As discussed in §2.5.2, it can be useful to estimate the resistance to guessing provided by a specific password x instead of analysing the guessing resistance of an entire distribution $\mathcal{X}$ from which x was drawn. This can be used for research to compare password choices between two groups with insufficient data to construct full distributions [163, 169, 273, 168]. It can also be used to build a proactive password-checker which indicates to a user the strength of a particular password during enrolment [37].

Current approaches[1] to estimating the strength of an individual password include using entropy estimation formulas [58, 321, 273], estimating the probability of the password in an model distribution [77, 310, 62], or estimating order in which the password would be guessed by cracking software [168].

In this chapter we formalise these approaches assuming we have complete knowledge of the underlying distribution $\mathcal{X}$. Our goal is to produce a *strength metric*, denoted $S_{\mathcal{X}}(x)$. Any strength metric is inherently dependent on our approximation $\mathcal{X}$ for the underlying distribution. Although we found that language difference did not make a major difference in guessing attacks on aggregate (§§8.2.2–8.2.3), it can make huge differences for individual passwords. For example, the password `tequiero` (which roughly translates from Spanish to English as `iloveyou`) is one of the top 100 passwords within the RockYou data set, but is over 25 times less common in the otherwise very similar BHeroes data set.

There are two key properties any useful strength metric S should have:

1. **Consistency for uniform distributions**: Similar to our preferred unit conversion for distribution-wide guessing metrics (§3.2.4), for a discrete uniform distribution $\mathcal{U}_N$ we

[1] One possible approach which does not appear to have been tried is to evaluate the *Kolmogorov complexity* [191] of a password, defined as the length of the shortest algorithm which can produce it. This approach is challenging because most passwords are so short they don't have a more compact algorithmic description.

want any strength metric to assign $\lg N$ bits to all events:

$$\forall_{x \in \mathcal{U}_N} \quad S_{\mathcal{U}_N}(x) = \lg N \tag{9.1}$$

2. **Monotonicity**: A strength metric should rate any event more weakly than events which are less common in the underlying distribution $\mathcal{X}$:

$$\forall_{x,x' \in \mathcal{X}} \quad p_x \geq p_{x'} \iff S_{\mathcal{X}}(x) \leq S_{\mathcal{X}}(x') \tag{9.2}$$

9.1 Strength metrics

We now formalise three sensible strength metrics which satisfy the above properties.

9.1.1 Probability metric

The simplest approach is to take the estimated probability p_x from $\mathcal{X}$ and estimate the size of a uniform distribution in which all elements have probability p_x:

$$S^{\mathrm{P}}_{\mathcal{X}}(x) = -\lg p_x \tag{9.3}$$

This is, in fact, the classic definition of the *self-information* of x (also called the *surprisal*), which is the underlying basis for Shannon entropy H_1 (§3.1.1). It is easy to see that for a randomly drawn value $x \xleftarrow{\mathrm{R}} \mathcal{X}$, the expected value of this metric is:

$$E\left[S^{\mathrm{P}}_{\mathcal{X}}(x) \middle| x \xleftarrow{\mathrm{R}} \mathcal{X}\right] = \sum_{i=1}^{|\mathcal{X}|} p_x \cdot -\lg p_x = H_1(\mathcal{X}) \tag{9.4}$$

Previous metrics which attempt to measure the probability of a password, for example using Markov models [77, 218, 62] or probabilistic context-free grammars [310], can be seen as attempts to approximate $S^{\mathrm{P}}_{\mathcal{X}}(x)$.

Applied to guessing, this model measures the (logarithmic) expected number of guesses before x is tried by an attacker who is guessing random values $x \xleftarrow{\mathrm{R}} \mathcal{X}$ with no memory. As discussed in the case of collision-entropy H_2 (§3.1.2), this attack model might apply if an attacker is guessing randomly according to the distribution to evade an intrusion detection system. For optimal sequential guessing attacks however, this metric has no direct relation.

9.1.2 Index metric

To estimate the strength of an individual event against an optimal guessing attack, the only relevant fact is the index i_x of x, that is, the number of items in $\mathcal{X}$ of greater probability than p_x. Using similar techniques to those in §3.2.4, we can convert this to an effective key-length

by considering the size of a uniform distribution which would, on average, require i_x guesses. This gives us a strength metric of:

$$S^{\mathrm{I}}_{\mathcal{X}}(x) = \lg\left(2 \cdot i_x - 1\right) \tag{9.5}$$

Intuitively, this metric is related to real-world attempts to measure the strength of an individual password by estimating when it would be guessed by password-cracking software [168].

A practical problem with this metric is that many events may have the same estimated probability. If we break ties arbitrarily, then in the case of the uniform distribution $\mathcal{U}_N$ the formula won't give $\lg N$ for all events. In fact, it won't even give $\lg N$ on average for all events, but instead $\approx \lg N - (\lg e - 1)$ (proved in §B.5).

A better approach for a sequence of j equiprobable events $x_i \cdots x_{i+j}$ where $p_i = \ldots = p_{i+j}$ is to assign index $\frac{i+j}{2}$ to all events when computing the index strength metric. This is equivalent to assuming an attacker will choose randomly given a set of remaining candidates with similar probabilities and it does give a value of $\lg N$ to all events in the uniform distribution.

This is slightly unsatisfactory however, as it means for a distribution $\mathcal{X} \approx \mathcal{U}_N$ which is "close" to uniform but with a definable ordering of the events, the average strength will appear to jump down by $(\lg e - 1) \approx 0.44$ bits.

A second problem is that the most probable event in $\mathcal{X}$ is assigned a strength of $\lg 1 = 0$. To satisfy our consistency requirement is a necessary limitation, since for $\mathcal{U}_1$ we must assign the single possible event a strength of $\lg 1 = 0$.

9.1.3 Partial guessing metric

The probability metric doesn't model a sequential guessing attack, while the index approach has a number of peculiarities. A better approach is to consider the minimum amount of work done per account by an optimal partial guessing attack which will compromise accounts using x. For example, if a user chooses the password `encryption`, an optimal attacker performing a sequential attack against the RockYou distribution will have broken 51.8% of accounts before guessing `encryption`. Thus, a user choosing the password `encryption` can expect to be safe from attackers who aren't aiming to compromise at least this many accounts, which takes $\tilde{G}_{0.518}$ work on average per account. We can turn this into a strength metric as follows:

$$S^{\mathrm{G}}_{\mathcal{X}}(x') = \tilde{G}_{\alpha_x}(\mathcal{X}) : \alpha_x = \sum_{i=1}^{i_x} p_i \tag{9.6}$$

Because $\tilde{G}_\alpha\left(\mathcal{U}_N\right) = \lg N$ for all α, the consistency property is trivially satisfied. Moreover, for "close" distributions $\mathcal{X} \approx \mathcal{U}_N$ where $|p_i - \frac{1}{N}| < \varepsilon$ for all i, we'll have $S^{\mathrm{G}}_{\mathcal{X}}(x_i) \to \lg N$ for all i as $\varepsilon \to 0$, unlike for S^{I} where strength will vary as long as there is any defined ordering.

As with $S^{\mathrm{I}}_{\mathcal{X}}$ though, we encounter the problem of of ordering for events with statistically indistinguishable probabilities. We'll apply the same tweak and give each event in a sequence of equiprobable events $x_i \cdots x_{i+j}$ the index $\frac{i+j}{2}$.

9.2 Estimation from a sample

Just like for distribution-wide metrics (§5), there are several issues when estimating strength metrics using an approximation for $\mathcal{X}$ obtained from a sample.

9.2.1 Estimation for unseen events

All of the above metrics are undefined for a previously unobserved event $x' \notin \mathcal{X}$. This is a result of our assumption that we have perfect information about $\mathcal{X}$. If not, we inherently need to rely on some approximation. As a consequence of the monotonicity requirement, $S(x' \notin \mathcal{X})$ must be $\geq \max(S(x \in \mathcal{X}))$. The naive formula for $S^{\mathrm{P}}_{\mathcal{X}}(x')$ assigns a strength estimate of ∞ though, which is not desirable.

We should therefore smooth our calculation of strength metrics by using some estimate $p(x') > 0$ even when $x' \notin \mathcal{X}$. Good-Turing techniques (§5.4) do not address this problem as they provide an estimate for the total probability of seeing a new event, but not the probability of a specific new event. A conservative approach is to add x' to $\mathcal{X}$ with a probability $\frac{1}{N+1}$, on the basis that if it is seen in practice in a distribution we're assuming is close to our reference $\mathcal{X}$, then we can treat x' as one additional observation about $\mathcal{X}$. This is analogous to the common heuristic of "add-one smoothing" in word frequency estimation [117]. This correction gives an estimate of $S^{\mathrm{P}}_{\mathcal{X}}(x) = \lg(N+1)$ for unobserved events.

For the index metric, this correction produces the smoothed estimate $S^{\mathrm{I}}_{\mathcal{X}}(x') = \lg 2N + 1$, an increase of roughly 1 bit, due to the aforementioned problem instability for a distribution $\mathcal{X} \approx \mathcal{U}_N$. For the partial guessing strength metric we have $S^{\mathrm{G}}_{\mathcal{X}}(x') \approx \tilde{G}_1(\mathcal{X})$, representing that guessing an unseen value requires at least an attacker willing to guess all known values.

All of these estimates are somewhat unsatisfactory because they don't allow us to distinguish between the estimated security of a new observation `encryption1` compared to `e5703572ae3c`, the latter of which intuitively seems much more difficult to guess. Solving this problem inherently relies on semantic evaluation of the newly-seen event, which is out of scope.

9.2.2 Stability of metrics

All of the proposed metrics will produce variable results for low-frequency events in a sample. The logarithmic nature of the estimators damps this problem to a large extent: if the hapax legomenon password `sapo26` occurred two more times in the RockYou data set, tripling its

observed frequency, its strength estimate would decrease by only 1.59, 2.22 and 2.55 bits for $S^{\mathrm{P}}_{\mathrm{RY}}$, $S^{\mathrm{I}}_{\mathrm{RY}}$, and $S^{\mathrm{G}}_{\mathrm{RY}}$, respectively.

It is straightforward to establish bounds on the worst case error when changing an event's observed probability from $p \to p' = p + \Delta p$. For S^{P}, the estimate can change from $\lg p$ to $\lg p'$, a difference of at most $\operatorname{abs}\left(\lg \frac{p'}{p}\right)$ bits.

For the index metric the worst-case scenario is that changing from $p \to p' = p + \Delta p$ changes the index by N if all other events have probability $p \leq p^* \leq p + \Delta p$. In this case, the maximum number of events in the distribution is $N = \frac{1}{\min(p,p')}$. This gives a worst-case change of $\lg(2N-1) - \lg 0 \approx \lg \frac{2}{\min(p,p')}$.

The worst-case change for S^{G} occurs in the same situation but is just $\lg N - \lg \frac{1}{p'} = \lg \frac{1}{p} - \lg \frac{1}{p'} = \operatorname{abs}\left(\lg \frac{p'}{p}\right)$, exactly as the case was for S^{P}. This is an attractive property of S^{G}: it offers worst-case stability equivalent to S^{P} while having a better connection to real guessing attacks.

However, in the special case where $\mathcal{X}$ is a Zipf distribution then the stability of S^{I} is equivalent to S^{P}. For a Zipf distribution each event's probability is roughly proportional to the inverse of its index in the distribution raised to a constant s:

$$p_x \propto \left(\frac{1}{i_x}\right)^s$$

Thus, if an event's probability increases by a constant factor k, its index should decrease by a factor of $k^{\frac{-1}{s}}$. This will decrease S^{P} by $\lg k$ bits and decrease S^{I} by $\frac{\lg k}{s}$ bits. For the classic Zipf distribution with $s \approx 1$, this means that changing an event's probability by k will affect $S^{\mathrm{I}}_{\mathcal{X}}$ and $S^{\mathrm{P}}_{\mathcal{X}}$ by exactly the same amount. While we reject the hypothesis that passwords are produced by a simple Zipfian distribution (§5.6), this is a rough justification for why we don't expect $S^{\mathrm{I}}_{\mathcal{X}}$ to be highly unstable in practice.

9.3 Application to individual passwords

Example values for the proposed strength metrics are given in Table 9.1 for passwords in the RockYou data set. Overall, the differences between S^{P}, S^{I}, and S^{G} are moderate with the exception of very common passwords, which receive significantly lower strength estimates by S^{I}. For much of the distribution, S^{G} provides estimates in between those of S^{P} and S^{I}, until the region of less-common passwords for which S^{G} is lower as it incorporates an attacker's ability to stop early upon success.

The fact that $S^{\mathrm{I}} < S^{\mathrm{P}}$ holds in every row is not a coincidence. Because the elements are ordered by probability, an event's index will always be lower in a skewed distribution than in a uniform distribution with identical events, so we will always have $S^{\mathrm{I}} \leq S^{\mathrm{P}}$.

The entropy estimation formula proposed by NIST [58] is shown for comparison as S^{NIST} (though note that S^{NIST} doesn't meet either of our desired mathematical criteria for a strength

x	$\lg(i_x)$	f_x	$S^{\mathrm{P}}_{\mathrm{RY}}$	$S^{\mathrm{I}}_{\mathrm{RY}}$	$S^{\mathrm{G}}_{\mathrm{RY}}$	S^{NIST}
`123456`	0	290729	6.81	0.00	6.81	14.0
`12345`	1	79076	8.69	1.58	7.46	12.0
`password`	2	59462	9.10	2.81	8.01	18.0
`rockyou`	3	20901	10.61	3.91	8.68	16.0
`jessica`	4	14103	11.17	4.95	9.42	16.0
`butterfly`	5	10560	11.59	5.98	10.08	19.5
`charlie`	6	7735	12.04	6.99	10.71	16.0
`diamond`	7	5167	12.62	7.99	11.30	16.0
`freedom`	8	3505	13.18	9.00	11.88	16.0
`letmein`	9	2134	13.90	10.00	12.48	16.0
`bethany`	10	1321	14.59	11.00	13.09	16.0
`lovers1`	11	739	15.43	12.00	13.74	22.0
`samanta`	12	389	16.35	13.00	14.42	16.0
`123456p`	13	207	17.27	14.00	15.13	22.0
`diving`	14	111	18.16	15.00	15.87	14.0
`flower23`	15	63	18.98	16.00	16.62	24.0
`scotty2hotty`	16	34	19.87	17.02	17.38	30.0
`lilballa`	17	18	20.79	18.01	18.13	18.0
`robbies`	18	9	21.79	19.06	18.93	16.0
`DANELLE`	19	5	22.64	19.96	19.62	22.0
`antanddeck06`	20	3	23.37	20.84	20.30	30.0
`babies8`	21	2	23.96	21.78	21.00	22.0
`sapo26`	22	1	24.96	24.00	22.44	20.0
`jcb82`	23	0	23.77	24.00	22.65	18.0

Table 9.1: Example strength estimates for a selection of passwords from the RockYou data set. The estimator S^{NIST} is calculated using the NIST entropy estimation formula [58].

metric). It fails for a few passwords which demonstrate the challenges of semantic evaluation: both `scotty2hotty` and `antanddeck06` score highly by S^{NIST} for being long and including digits. Neither is particularly strong, however: `scotty2hotty` is a professional wrestler, while `antanddeck06` is based on the name of a British comedy show. In contrast `sapo26` is much shorter and rated 10 bits lower by S^{NIST}, but doesn't have a well-known real-world meaning.

Because we listed passwords in order of exponentially increasing index, we can test the Zipfian relationship on the difference between S^{P} and S^{I} using the data by comparing the ratio of differences for successive passwords x_2, x_1 in Table 9.1:

$$s \approx \frac{S^{\mathrm{P}}_{\mathrm{RY}}(x_{i+1}) - S^{\mathrm{P}}_{\mathrm{RY}}(x_i)}{S^{\mathrm{I}}_{\mathrm{RY}}(x_{i+1}) - S^{\mathrm{I}}_{\mathrm{RY}}(x_i)}$$

For successive rows of the table, we get estimates for s ranging from 0.34 to 1.37. The average estimate, however, is $s = 0.76$, almost identical to the estimate we would get by computing s using only the first and last row of the table. In Figure 5.2, we computed a power-law fit to the rank data with exponent $2.11 \leq a \leq 2.66$. Using the equivalence between the power-law and Zipf formulation that $a = 1 + \frac{1}{s}$, we would estimate $a = 2.31$ given $s = 0.76$. This is not a sound way of computing a Zipfian fit for the data set s in general, but the fact that it is roughly consistent supports our hypothesis that S^{I} will be stable for realistic distributions which follow a (very rough) power-law approximation.

9.4 Application to small data sets

A second application of strength metrics is to estimate the average strength given only a very small sample for which distribution-wide statistics (§3) can't be computed. This method can only be accurate for data sets which are approximately drawn from the same population as the base distribution, though this limitation is equally true of password cracking (§2.5.1) or semantic evaluation (§2.5.2).

If we interpret the small set of passwords as a sample from some larger distribution, we need to reason about the expected value of each strength metric. We've already shown in Equation 9.4 that $E\left[S^{\mathrm{P}}_{\mathcal{X}}(x) \middle| x \xleftarrow{\mathrm{R}} \mathcal{X}\right] = H_1(\mathcal{X})$. The expected value of S^{I} was too complicated to compute directly even for a uniform distribution. Similarly, the expected value of S^{G} is:

$$E\left[S^{\mathrm{G}}_{\mathcal{X}}(x) \middle| x \xleftarrow{\mathrm{R}} \mathcal{X}\right] = \int_0^1 \tilde{G}_\alpha(\mathcal{X}) \mathrm{d}\alpha \tag{9.7}$$

which doesn't appear to admit a simple analytic formula. Instead, we can only compute $E\left[S^{\mathrm{I}}_{\mathcal{X}}(x) \middle| x \xleftarrow{\mathrm{R}} \mathcal{X}\right]$ and $E\left[S^{\mathrm{G}}_{\mathcal{X}}(x) \middle| x \xleftarrow{\mathrm{R}} \mathcal{X}\right]$ directly for our reference distribution $\mathcal{X}$ and use this as a benchmark for comparison against a smaller distribution.

In Table 9.2 a variety of small password data sets for which cleartext passwords are available are evaluated using the RockYou data set as a baseline. None of the statistical metrics are obviously superior, though S^{G} is typically in between the values produced by the other two.

The NIST formula produces more plausible results when averaged than for individual passwords (as in §9.3), correctly ranking the 2011 Twitter blacklist as much weaker than the other lists (though not as weak as the statistical estimates). The NIST formula also plausibly rates the foreign-language Hebrew data set lower than the statistical estimates, as it doesn't assume the passwords are in English like using RockYou as a baseline implicitly does.

In the myBart data set about two-thirds of users retained site-assigned random passwords. This set was rated highly by all of the metrics, being recognised inadvertently by the NIST formula because the site-assigned passwords always contained a number.

The largest difference in the rankings occurred for the Hotmail and MySpace data sets, which produced indistinguishable statistical estimates but differed by over 4 bits by the NIST

Dataset	M	% seen	$S^{\mathrm{P}}_{\mathrm{RY}}$	$S^{\mathrm{I}}_{\mathrm{RY}}$	$S^{\mathrm{G}}_{\mathrm{RY}}$	S^{NIST}
RockYou (baseline)	—	100.0%	21.15	18.79	18.75	19.82
small password sets						
70yx (sampled)	1000	34.0%	22.28	21.24	21.52	20.21
Fox	369	68.8%	20.95	18.99	19.33	19.28
Hebrew	1307	50.3%	21.25	19.63	20.14	17.46
Hotmail	11576	57.6%	21.82	20.29	20.43	18.21
myBart	2007	19.0%	22.93	22.37	22.54	23.53
MySpace	50546	59.5%	21.64	20.02	20.19	22.53
NATO-Books	11822	50.9%	21.66	20.17	20.47	19.35
Sony-BMG	41024	61.3%	20.93	19.10	19.53	19.87
malware dictionaries						
Conficker	190	96.8%	16.99	13.60	15.07	16.51
Morris	445	94.4%	18.62	15.68	16.56	15.27
blacklists						
Twitter-2010	404	7.9%	23.16	22.86	23.02	15.30
Twitter-2011	429	99.8%	15.11	11.31	13.46	15.27

Table 9.2: Average strength estimates for small lists of leaked passwords. Details of the data sets are provided in §D. The NIST entropy estimation formula [58] is listed as S^{NIST}.

formula. Examining the passwords, it appears that a good portion of the MySpace data was collected under a policy (§2.2.4) mandating non-alphabetic characters: `password1` is the most popular password, over twice as popular as `password`, and most of the other top passwords include a number. Popular numeric passwords such as `123456` appear to have been banned under some of the collection rules, as they are less common than variants like `123456a`. The Hotmail data set, on the other hand, appears to have had no restrictions. Because the NIST formula awards a constant 6 points to passwords with a mix of numbers and letters, the MySpace complexity policy significantly raises S^{NIST}. However, the statistical estimators suggest these passwords may not actually be much stronger by this policy as a large number of users simply append a digit (usually a `1` or `0`) to a weak password. In this case, statistical strength metrics are less influenced by the effects of complexity requirements.

The NIST formula also struggled to recognise datasets of explicitly weak passwords. For example, it considers the Conficker password dictionary to contain stronger passwords than the the Morris password dictionary, though our analysis in Table 8.2 suggested that the Conficker list is a better attack dictionary (containing much weaker passwords). Similarly, the two versions of the Twitter blacklist are rated similarly by the NIST formula, but the statistical metrics identify the 2011 version as a vast improvement (again in that it contains weaker passwords).

It's quite possible the correct conclusion is that typed-in passwords are fundamentally hopeless as a means of authenticating users.

—Barry Shein via Usenet, 1989 [160]

Chapter 10

Conclusions and perspectives

The goals of this dissertation were to introduce a sound framework for analysing guessing attacks and apply it to large real-world data sets to accurately estimate guessing difficulty. To the first goal, our partial guessing metrics are a significant improvement over Shannon entropy or guesswork, both of which are difficult to estimate from a sample and don't provide meaningful information for practical distributions. It is also easy to find examples where entropy estimation formulas misinterpret password semantics and produce misleading results.

The primary difficulty with using statistical guessing metrics is collecting a sample of sufficient size. Even our largest-ever data set of nearly 70 million passwords proved inadequate for evaluating brute-force attacks expending a significant amount of effort to break a high proportion $\alpha > 50\%$ of accounts. It remains an open question exactly how difficult it is to guess passwords from the long tail of the distribution.

Larger-scale cracking evaluations might help to answer this question approximately, though designers of password cracking tools face essentially the same problem: when guessing very rare passwords for which statistical metrics break down, it is difficult to properly tune the order of candidate passwords due to the lack of data. Thus, even when cracking tools can break a larger proportion of passwords, the divergence from an optimal attack will be unknown and increasing with α.

Another approach would be to find a better model distribution for passwords. The zero-truncated Sichel/Poisson distribution produces empirically accurate estimations of guessing metrics for $\alpha \lessapprox 0.5$, but we have no way of knowing how good the fit is for less likely passwords and there is insufficient a priori justification that this is an appropriate model.

Further research on the neurophysiology of generating and remembering secrets might give us more confidence in a model distribution, but we doubt that any simple mathematical model can fully explain password selection. In the case of PINs (§4) we identified a mixture

distribution with individuals choosing very different strategies, including random selection for a significant proportion of the population. For passwords, as well, at least some observed values are machine-chosen pseudorandom strings (§3.1.3). Other users may choose smaller random strings for passwords or pseudorandom values such as old phone numbers. Because passwords have few limitations compared to PINs, adapting our regression model to estimate the mixture of password selection strategies would be challenging.

Most proposals to mathematically model passwords adapt linguistic approaches like power-law distributions or context-free grammars [310]. Yet the extent to which natural language grammar is modeled by context-free grammars remains controversial after decades of linguistic research [234], as does the extent to which natural language distributions of words can be modeled by power-law distributions [23, 69]. We conjecture that the distribution of human-chosen secrets is at least as complex as natural language, making it very difficult to find a complete model.

As to our second goal, we have endeavoured to compare the guessing difficulty of all commonly used distributions of human-chosen secrets in a universal framework, as plotted in Figure 10.1. There appears to be a natural ordering with passwords ranging from 10–20 bits of security, personal knowledge questions being about equivalent against online guessing and significantly weaker against brute-force attacks, and PINs providing some security against very limited online attacks but inevitably very little against brute force. Graphical and mnemonic passwords appear stronger based on limited cracking attempts, though not significantly better than early password cracking estimates. This figure should serve as a reference for security engineers when designing new systems that incorporate human-chosen distributions.

The numbers on passwords, when converted to bits, appear hopelessly weak to trained cryptographers who advocate a minimum of 80 bits of security and employ 128 bits for mundane applications. Our results are consistent with decades of research suggesting passwords are wholly inadequate for high-security applications. Clearly though, passwords have survived and probably thwarted millions of opportunistic guessing attacks. There is no magic number of bits which would make passwords or any other distribution "secure." Even numeric PINs can be a useful component in larger security systems. It is essential though that, when human-chosen secrets must be used, security engineers understand the cost of guessing attacks and design layers of resilience to survive them.

What's more discouraging about available password data is how little security varies between sites, between different demographics of users, or even between users with seemingly different security requirements. It's possible that users' energy to choose passwords successfully has been eroded by the huge number of passwords they are asked to create [121] and some of the insecurity observed is an artifact of this higher-level usability failure. It's also possible that humans are inherently unable to collectively produce a strong distribution of secrets even when strongly motivated.

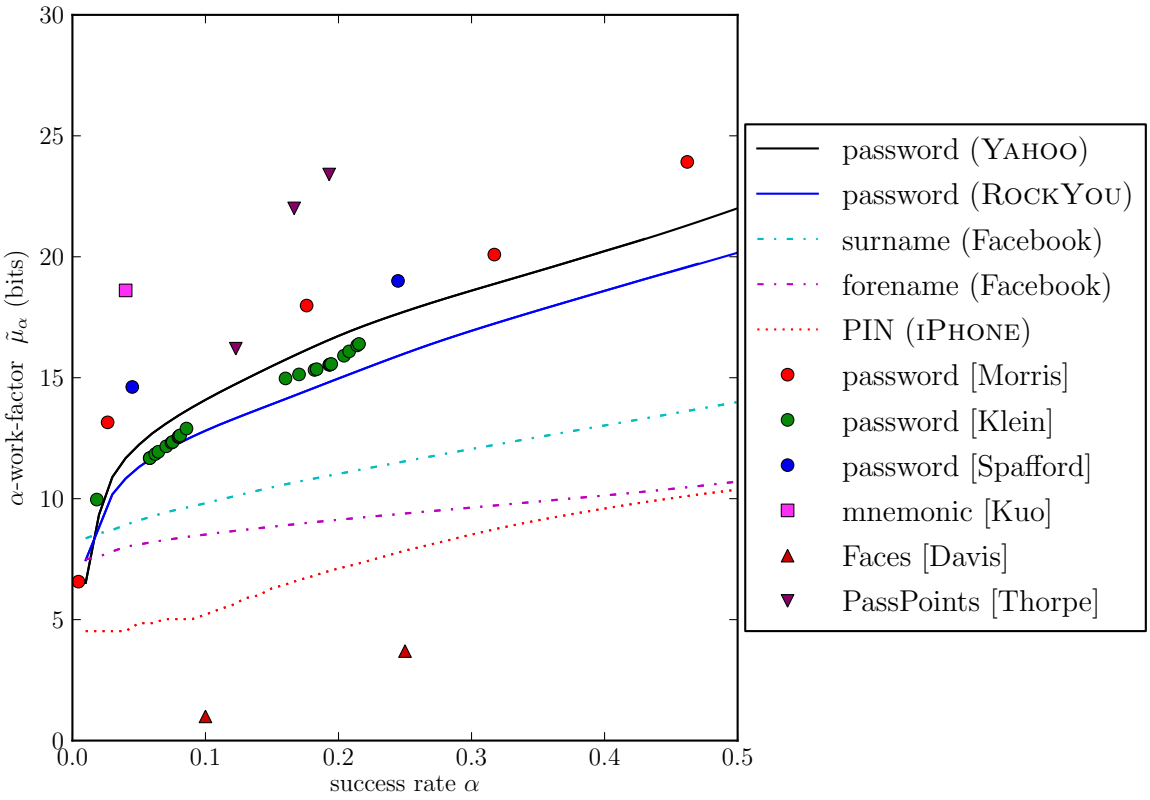


Figure 10.1: Meta-comparison of different human-chosen distributions.

Future designers of authentication schemes must plan for human choice to introduce a skewed distribution and never assume user behaviour can be approximated as a random choice from a fixed set of possibilities. While it is difficult to collect significant statistics for newly proposed schemes to compute the preferred metric $\tilde{G}_\alpha$, the work of Thorpe and van Oorschot [294, 301] provides a good example of actively seeking to measure weak subspaces within new human-chosen distributions. Estimated guessing difficulty should be reported in a standard format of either (α, μ) to represent the proportion α of users falling into a subspace of size μ.

Finally, it appears from Figure 10.1 and all of the other empirical estimates presented in this dissertation that the level of security provided by current systems is so low that it might be worth returning to machine-chosen secrets for security-critical applications (§2.3.2). Providing $H_\infty = 30$ bits of security against all attacks would be a vast improvement for the majority of users and can be represented by a 9-digit number or a sequence of 3 common English words. Perhaps previous proposals have failed by attempting to provide much higher-cryptographic levels of security at which point memorisation is too difficult. Accepting the security limits of user-chosen secrets and striving to gradually improve on them may be the best approach to finally begin moving human-computer authentication forward.

Bibliography

[1] John the Ripper. `http://www.openwall.com/john/`.

[2] Data Encryption Standard. Technical Report FIPS PUB 46, National Institute of Standards and Technology, 1977.

[3] Password Usage. *United States Federal Information Processing Standards Publication 112*, 1985.

[4] Automated Password Generator (APG). Technical Report FIPS PUB 181, National Institute of Standards and Technology, 1993.

[5] Pubcookie Design Specifications. `http://www.pubcookie.org/docs/specs.html`, 2003.

[6] *EMV Integrated Circuit Card Standard for Payment Systems version 4.2.* EMVco, 2008.

[7] Symantec Report on the Underground Economy, 2008. Symantec Corporation.

[8] Verified by Visa. `www.visa.com/verifiedbyvisa/`, 2010.

[9] *ISO 9564:2011 Financial services—Personal Identification Number (PIN) management and security.* International Organisation for Standardisation, 2011.

[10] Microsoft Passport, 2011. `https://www.passport.net`.

[11] PassWindow. `http://www.passwindow.com`, 2011.

[12] Data Breach Investigative Report. Verizon, Inc., 2012.

[13] *The Unicode Standard Version 6.1.* The Unicode Consortium, 2012.

[14] Anne Adams and Martina Angela Sasse. Users are Not the Enemy. *Communications of the ACM*, 42(12):40–46, 1999.

[15] Anne Adams, Martina Angela Sasse, and Peter Lunt. Making Passwords Secure and Usable. In *HCI 97: Proceedings of HCI on People and Computers XII*, pages 1–19, London, UK, 1997. Springer-Verlag.

[16] Ben Adida. Beamauth: Two-Factor Web Authentication with a Bookmark. In *CCS '07: Proceedings of the 14th ACM Conference on Computer and Communications Security*, pages 48–57, New York, NY, USA, 2007. ACM.

[17] Ben Adida. EmID: Web Authentication by Email Address. In *W2SP '08: Proceedings of Web 2.0 Security and Privacy Workshop*, 2008.

[18] Petar S. Aleksic and Aggelos K. Katsaggelos. Audio-Visual Biometrics. In *Proceedings of the IEEE*, volume 94, pages 2025–2044, 2006.

[19] Mansour Alsaleh, Mohammad Mannan, and P.C. van Oorschot. Revisiting Defenses Against Large-Scale Online Password Guessing Attacks. *IEEE Transactions on Dependable and Secure Computing*, 9(1):128–141, 2012.

[20] James P. Anderson. Information Security in a Multi-User Computer Environment. volume 12 of *Advances in Computers*, pages 1–36. Elsevier, 1972.

[21] Ross J. Anderson. Cryptography and Competition Policy — Issues with 'Trusted Computing'. In *PODC '03: Proceedings of the 22nd Annual Symposium on Principles of Distributed Computing*, pages 3–10, New York, NY, USA, 2003. ACM.

[22] Ross J. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley, New York, 2$^{\text{nd}}$ edition, 2008.

[23] Harald R. Baayen. *Word Frequency Distributions*. Text, Speech and Language Technology. Springer, 2001.

[24] Lucas Ballard, Seny Kamara, and Michael K. Reiter. The Practical Subtleties of Biometric Key Generation. In *Proceedings of the 17th USENIX Security Symposium*, pages 61–74, Berkeley, CA, USA, 2008.

[25] Lucas Ballard, Daniel Lopresti, and Fabian Monrose. Evaluating the Security of Handwriting Biometrics. In *10th International Workshop on Frontiers in Handwriting Recognition*. Université de Rennes 1, Suvisoft, 2006.

[26] Davide Balzarotti, Marco Cova, and Giovanni Vigna. ClearShot: Eavesdropping on Keyboard Input from Video. In *SP '08: Proceedings of the 2008 IEEE Symposium on Security and Privacy*, pages 170–183, Washington, DC, USA, 2008. IEEE Computer Society.

[27] Gregory V. Bard. Spelling-Error Tolerant, Order-Independent Pass-Phrases via the Damerau-Levenshtein String-Edit Distance Metric. In *ACSW '07: Proceedings of the 5th Australasian Symposium on ACSW Frontiers*, volume 68, pages 117–124, Darlinghurst, Australia, 2007. Australian Computer Society, Inc.

[28] Ben F. Barton and Marthalee S. Barton. User-friendly password methods for computer-mediated information systems. *Computers & Security*, 3:186–195, 1984.

[29] Bernardo Bátiz-Lazo and Robert J.K. Reid. The Development of Cash-Dispensing Technology in the UK. *IEEE Annals of the History of Computing*, 33:32–45, 2011.

[30] J. Beirlant, E. J. Dudewicz, L. Györfi, and E. C. Meulen. Nonparametric Entropy Estimation: An Overview. *International Journal of Mathematics, Statistics and Science*, 6(1), 1997.

[31] Steven M. Bellovin and Michael Merritt. Encrypted Key Exchange: Password-Based Protocols SecureAgainst Dictionary Attacks. In *SP '92: Proceedings of the 1992 IEEE Symposium on Security and Privacy*, pages 72–84, Washington, DC, USA, 1992. IEEE Computer Society.

[32] Steven M. Bellovin and Michael Merritt. Augmented Encrypted Key Exchange: A Password-Based Protocol Secure Against Dictionary Attacks and Password File Compromise. In *CCS '93: Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 244–250, New York, NY, USA, 1993. ACM.

[33] F. Bergadano, B. Crispo, and G. Ruffo. Proactive Password Checking with Decision Trees. In *CCS '97: Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 67–77, New York, NY, USA, 1997. ACM.

[34] Vittorio Bertocci, Garrett Serack, and Caleb Baker. *Understanding Windows Cardspace: An Introduction To the Concepts and Challenges of Digital Identities*. Addison-Wesley Professional, 1st edition, 2007.

[35] Robert Biddle, Sonia Chiasson, and P.C. van Oorschot. Graphical Passwords: Learning from the First Twelve Years. Technical Report TR-11-01, Carleton University, 2011.

[36] Matt Bishop. A Proactive Password Checker. Technical report, Hanover, NH, USA, 1990.

[37] Matt Bishop and Daniel V. Klein. Improving System Security via Proactive Password Checking. *Computers & Security*, 14(3):233–249, 1995.

[38] Elizabeth Ligon Bjork and Robert Bjork, editors. *Memory: Handbook of Perception and Cognition*. Academic Press, Inc., 1998.

[39] Burton H. Bloom. Space/Time Trade-Offs in Hash Coding with Allowable Errors. *Communications of the ACM*, 13(7):422–426, 1970.

[40] Mike Bond. Comments on grIDsure authentication. `http://www.cl.cam.ac.uk/~mkb23/research/GridsureComments.pdf`, 2008.

[41] Mike Bond and Piotr Zieliński. Decimalisation table attacks for PIN cracking. Technical Report UCAM-CL-TR-560, University of Cambridge, 2003.

[42] Joseph Bonneau. Getting web authentication right: a best-case protocol for the remaining life of passwords. In *19th International Workshop on Security Protocols*, 2011.

[43] Joseph Bonneau. Statistical metrics for individual password strength. In *20th International Workshop on Security Protocols*, 2012.

[44] Joseph Bonneau. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *SP '12: Proceedings of the 2012 IEEE Symposium on Security and Privacy*, 2012.

[45] Joseph Bonneau, Cormac Herley, Paul C. van Oorschot, and Frank Stajano. The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In *SP '12: Proceedings of the 2012 IEEE Symposium on Security and Privacy*, 2012.

[46] Joseph Bonneau, Mike Just, and Greg Matthews. What's in a name? Evaluating statistical attacks against personal knowledge questions. In *FC '10: The 14th International Conference on Financial Cryptography and Data Security*. Springer-Verlag, 2010.

[47] Joseph Bonneau and Sören Preibusch. The password thicket: technical and market failures in human authentication on the web. In *WEIS '10: Proceedings of the 9th Workshop on the Economics of Information Security*, 2010.

[48] Joseph Bonneau, Sören Preibusch, and Ross Anderson. A birthday present every eleven wallets? The security of customer-chosen banking PINs. In *FC '12: The 16th International Conference on Financial Cryptography and Data Security*. Springer-Verlag, 2012.

[49] Joseph Bonneau and Ekaterina Shutova. Linguistic properties of multi-word passphrases. In *USEC '12: Workshop on Usable Security*, 2012.

[50] Serdar Boztas. Entropies, Guessing, and Cryptography. Technical Report 6, Department of Mathematics, Royal Melbourne Institute of Technology, 1999.

[51] John Brainard, Ari Juels, Ronald L. Rivest, Michael Szydlo, and Moti Yung. Fourth-Factor Authentication: Somebody You Know. In *CCS '06: Proceedings of the 13th ACM Conference on Computer and Communications Security*, pages 168–178, New York, NY, USA, 2006. ACM.

[52] Thorsten Brantz and Alex Franz. The Google Web 1T 5-gram corpus. Technical Report LDC2006T13, Linguistic Data Consortium, 2006.

[53] Peter Bright. RSA finally comes clean: SecurID is compromised. *Ars Technica*, 2011.

[54] Sacha Brostoff and Angela Sasse. "Ten strikes and you're out": Increasing the number of login attempts can improve password usability. In *Proceedings of CHI 2003 Workshop on HCI and Security Systems*. John Wiley, 2003.

[55] Sacha Brostoff and M. Angela Sasse. Are Passfaces More Usable Than Passwords? A Field Trial Investigation. In *People and Computers XIV: Usability or Else!: Proceedings of HCI 2000*, 2000.

[56] Daniel R. L. Brown. Prompted User Retrieval of Secret Entropy: The Passmaze Protocol. Cryptology ePrint Archive, Report 2005/434, 2005. `http://eprint.iacr.org/`.

[57] Julie Bunnell, John Podd, Ron Henderson, Renee Napier, and James Kennedy-Moffat. Cognitive, associative and conventional passwords: Recall and guessing rates. *Computers & Security*, 16(7):629–641, 1997.

[58] William E. Burr, Donna F. Dodson, and W. Timothy Polk. Electronic Authentication Guideline. *NIST Special Publication 800-63*, 2006.

[59] Eric Butler. Firesheep, 2011. `codebutler.com/firesheep`.

[60] Christian Cachin. *Entropy Measures and Unconditional Security in Cryptography*. PhD thesis, ETH Zürich, 1997.

[61] John Andrew Campbell, Kay Bryant, Mary-Anne Williams Sue Williams Steve Elliot, Carol Pollard, and Carol Pollard. Password composition and Security: An Exploratory Study of User Practice. 2004.

[62] Claude Castelluccia, Markus Dürmuth, and Daniele Perito. Adaptive Password-Strength Meters from Markov Models. In *NDSS '12: Proceedings of the Network and Distributed System Security Symposium*, 2012.

[63] Joseph A. Cazier and B. Dawn Medlin. Password Security: An Empirical Investigation into E-Commerce Passwords and Their Crack Times. *Information Systems Security*, 15(6):45–55, 2006.

[64] William Cheswick. Johnny Can Obfuscate: Beyond Mother's Maiden Name. In *Proceedings of the 1st USENIX Workshop on Hot Topics in Security*, pages 31–36, Berkeley, CA, USA, 2006. USENIX Association.

[65] Sonia Chiasson, Alain Forget, Robert Biddle, and P.C. van Oorschot. Influencing Users Towards Better Passwords: Persuasive Cued Click-Points. In *BCS-HCI '08: Proceedings*

of the 22nd British HCI Group Annual Conference on HCI 2008, pages 121–130, Swinton, UK, 2008. British Computer Society.

[66] Sonia Chiasson, Alain Forget, Elizabeth Stobert, P.C. van Oorschot, and Robert Biddle. Multiple Password Interference in Text Passwords and Click-Based Graphical Passwords. In *CCS '09: Proceedings of the 16th ACM Conference on Computer and Communications Security*, pages 500–511, New York, NY, USA, 2009. ACM.

[67] Sonia Chiasson, P.C. van Oorschot, and Robert Biddle. A Usability Study and Critique of Two Password Managers. In *Proceedings of the 15th USENIX Security Symposium*, 2006.

[68] Angelo Ciaramella, Paolo D'Arco, Alfredo De Santis, Clemente Galdi, and Roberto Tagliaferri. Neural Network Techniques for Proactive Password Checking. *IEEE Transactions on Dependable and Secure Computing*, 3:327–339, 2006.

[69] Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. Power-Law Distributions in Empirical Data. *SIAM Review*, 51:661–703, 2009.

[70] Michael Comer. Password breaking. *Computer Fraud & Security Bulletin*, 4(3):7–8, 1981.

[71] Richard M. Conlan and Peter Tarasewich. Improving Interface Designs to Help Users Choose Better Passwords. In *CHI '06: Extended Abstracts on Human Factors in Computing Systems*, pages 652–657, New York, NY, USA, 2006. ACM.

[72] Microsoft Corporation. Security configuration guidance support, 2010. `http://support.microsoft.com/kb/885409`.

[73] Baris Coskun and Cormac Herley. Can "Something You Know" Be Saved? In *ISC '08: Proceedings of the 11th International Conference on Information Security*, pages 421–440, Berlin, Heidelberg, 2008. Springer-Verlag.

[74] Johanna Bromberg Craig, Wes Craig, Kevin McGowan, and Jarod Malestein. The Cosign Web Single Sign-On Scheme. `http://cosign.sourceforge.net/media/cosignscheme2006a.rtf`, 2006.

[75] Nik Cubrilovic. The Anatomy Of The Twitter Attack. *TechCrunch*, July 2009.

[76] John Daugman. New Methods in Iris Recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 37(5):1167–1175, 2007.

[77] Chris Davies and Chris Ganesan. BApasswd: A New Proactive Password Checker. In *Proceedings of the 16th National Computer Security Conference*, 1993.

[78] Darren Davis, Fabian Monrose, and Michael K. Reiter. On User Choice in Graphical Password Schemes. In *Proceedings of the 13th USENIX Security Symposium*, 2004.

[79] A. De Santis, A.G. Gaggia, and U. Vaccaro. Bounds on Entropy in a Guessing Game. *IEEE Transactions on Information Theory*, 47(1):468–473, 2001.

[80] Khosrow Dehnad. A simple way of improving the login security. *Computers & Security*, 8:607–611, 1989.

[81] Matteo Dell'Amico, Pietro Michiardi, and Yves Roudier. Password Strength: An Empirical Analysis. In *INFOCOM'10: Proceedings of the 29th Conference on Information Communications*, pages 983–991. IEEE, 2010.

[82] Dorothy E. Denning. An Intrusion-Detection Model. *IEEE Transactions on Software Engineering*, 13(2):222–232, 1987.

[83] Dorothy E. Denning and Peter J. Denning. The Tracker: A Threat to Statistical Database Security. *ACM Transactions on Database Systems*, 4:76–96, 1979.

[84] Martin M.A. Devillers. Analyzing Password Strength. Technical report, Radboud University Nijmegen, 2010.

[85] Arkajit Dey and Stephen Weis. PseudoID: Enhancing Privacy for Federated Login. In *HotPETS '10: Hot Topics in Privacy Enhancing Technologies*, 2010.

[86] R. Dhamija and L. Dusseault. The Seven Flaws of Identity Management: Usability and Security Challenges. *IEEE Security & Privacy Magazine*, 6(2):24–29, 2008.

[87] Rachna Dhamija and Adrian Perrig. Déjà vu: A user study using images for authentication. In *Proceedings of the 9th USENIX Security Symposium*, Berkeley, CA, USA, 2000. USENIX Association.

[88] Rachna Dhamija, J. D. Tygar, and Marti Hearst. Why Phishing Works. In *CHI '06: Proceedings of the 24th ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 581–590, New York, NY, USA, 2006. ACM.

[89] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) protocol, 2006. RFC 4346.

[90] Paul Dourish, E. Grinter, Jessica Delgado de la Flor, and Melissa Joseph. Security in the Wild: User Strategies for Managing Security as an Everyday, Practical Problem. *Personal Ubiquitous Computing*, 8(6):391–401, 2004.

[91] Peter J. Downey. *Multics Security Evaluation: Password and File Encryption Techniques*. Ft. Belvoir Defense Technical Information Center, 1977.

[92] Sever S. Dragomir and Serdar Boztas. Some Estimates of the Average Number of Guesses to Determine a Random Variable. In *Proceedings of the 1997 IEEE International Symposium on Information Theory*, page 159, 1997.

[93] Saar Drimer, Steven J. Murdoch, and Ross Anderson. Optimised to Fail: Card Readers for Online Banking. In *FC '09: The 13th International Conference on Financial Cryptography and Data Security*, Berlin, Heidelberg, 2009.

[94] Cynthia Dwork. Differential Privacy. In *Automata, Languages and Programming*, volume 4052, pages 1–12. Springer Berlin / Heidelberg, 2006. 10.1007/11787006_1.

[95] Bradley Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall/CRC, 1st edition, 1993.

[96] Serge Egelman, Joseph Bonneau, Sonia Chiasson, David Dittrich, and Stuart Schechter. Its Not Stealing If You Need It: On the ethics of performing research using public data of illicit origin (panel discussion). In *WECSR '12: The 3rd Workshop on Ethics in Computer Security Research*, 2012.

[97] Carl Ellison, Chris Hall, Randy Milbert, and Bruce Schneier. Protecting Secret Keys with Personal Entropy. *Journal of Future Generation Computer Systems*, 16(4):311–318, 2000.

[98] Laura Falk, Atul Prakash, and Kevin Borders. Analyzing Websites for User-Visible Security Design Flaws. In *SOUPS '08: Proceedings of the 4th Symposium on Usable Privacy and Security*, pages 117–126, New York, NY, USA, 2008. ACM.

[99] David C. Feldmeier and Philip R. Karn. UNIX Password Security—Ten Years Later. In *CRYPTO '89: Proceedings of the 9th Annual International Conference on Advances in Cryptology*, pages 44–63, London, UK, 1990. Springer-Verlag.

[100] Adrienne Porter Felt, Matthew Finifter, Erika Chin, Steve Hanna, and David Wagner. A Survey of Mobile Malware in the Wild. In *SPSM '11: Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, pages 3–14, New York, NY, USA, 2011. ACM.

[101] R. A. Fisher, A. S. Corbet, and C. B. Williams. The relation between the number of species and the number of individuals in a random sample of an animal population. *Journal of Animal Ecology*, 12:42–58, 1943.

[102] Dinei Florêncio and Cormac Herley. KLASSP: Entering Passwords on a Spyware Infected Machine Using a Shared-Secret Proxy. In *ACSAC '06: Proceedings of the 22nd Annual Computer Security Applications Conference*, pages 67–76, Washington, DC, USA, 2006. IEEE Computer Society.

[103] Dinei Florêncio and Cormac Herley. A Large-Scale Study of Web Password abits. In *WWW '07: Proceedings of the 16th International Conference on the World Wide Web*, pages 657–666. ACM, 2007.

[104] Dinei Florêncio and Cormac Herley. One-Time Password Access to Any Server without Changing the Server. In *ISC '08: Proceedings of the 11th International Conference on Information Security*, pages 401–420, Berlin, Heidelberg, 2008. Springer-Verlag.

[105] Dinei Florêncio and Cormac Herley. Where Do Security Policies Come From? In *SOUPS '10: Proceedings of the 6th Symposium on Usable Privacy and Security*. ACM, 2010.

[106] M. Font, X. Puig, and J. Ginebra. A Bayesian analysis of frequency count data. *Journal of Statistical Computation and Simulation*, 2011.

[107] Mozilla Foundation. BrowserID, 2012. `browserid.org/`.

[108] Mozilla Foundation. Firefox Password Manager. `wiki.mozilla.org/Firefox:Password_Manager`, 2012.

[109] Wendy R. Fox and Gabriel W. Lasker. The Distribution of Surname Frequencies. *International Statistical Review*, pages 81–87, 1983.

[110] A.D. Frankel and M. Maheswaran. Feasibility of a Socially Aware Authentication Scheme. In *CCNC '09: Proceedings of the 6th IEEE Consumer Communications and Networking Conference, 2009*, pages 1–6, 2009.

[111] John Franks, Phillip M. Hallam-Baker, Jeffery L. Hostetler, Scott D. Lawrence, Paul J. Leach, Ari Luotonen, and Lawrence C. Stewart. HTTP Authentication: Basic and Digest Access Authentication, 1999. RFC 2617.

[112] Niklas Frykholm and Ari Juels. Error-Tolerant Password Recovery. In *CCS '08: Proceedings of the 8th ACM Conference on Computer and Communications Security*, pages 1–9, New York, NY, USA, 2001. ACM.

[113] Kevin Fu, Emil Sit, Kendra Smith, and Nick Feamster. Dos and Don'ts of Client Authentication on the Web. In *Proceedings of the 10th USENIX Security Symposium*, pages 19–43, Berkeley, CA, USA, 2001. USENIX Association.

[114] Steven Furnell. Authenticating ourselves: will we ever escape the password? *Network Security*, 2005(3):8–13, 2005.

[115] Steven Furnell. An Assessment of Website Password Practices. *Computers & Security*, 26(7-8):445–451, 2007.

[116] Eran Gabber, Phillip B. Gibbons, Yossi Matias, and Alain J. Mayer. How to Make Personalized Web Browising Simple, Secure, and Anonymous. In *FC '97: Proceedings*

of the 1st International Conference on Financial Cryptography, pages 17–32, London, UK, 1997. Springer-Verlag.

[117] William A. Gale and Geoffrey Sampson. Good-Turing Frequency Estimation Without Tears. *Journal of Quantitative Linguistics*, 2(3):217–237, 1995.

[118] Ravi Ganesan and Chris Davies. A New Attack on Random Pronounceable Password Generators. In *Proceedings of the 17th NIST-NCSC National Computer Security Conference*, 1994.

[119] Simson L. Garfinkel. Email-Based Identification and Authentication: An Alternative to PKI? *IEEE Security & Privacy Magazine*, 1(6):20–26, 2003.

[120] Morrie Gasser. A Random Word Generator for Pronounceable Passwords. Technical Report MTR-3006, MITRE Corp, 1975.

[121] Shirley Gaw and Edward W. Felten. Password Management Strategies for Online Accounts. In *SOUPS '06: Proceedings of the 2nd Symposium on Usable Privacy and Security*, pages 44–55, New York, NY, USA, 2006. ACM.

[122] John Gilmore. *Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design.* Electronic Frontier Foundation, 1998.

[123] Oded Goldreich and Yehuda Lindell. Session-Key Generation using Human Passwords Only. In *CRYPTO '01: Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, pages 408–432, London, UK, UK, 2001. Springer-Verlag.

[124] Philippe Golle and David Wagner. Cryptanalysis of a Cognitive Authentication Scheme (Extended Abstract). In *SP '07: Proceedings of the 2007 IEEE Symposium on Security and Privacy*, pages 66–70, Washington, DC, USA, 2007. IEEE Computer Society.

[125] I. J. Good. The Population Frequencies of Species and the Estimation of Population Parameters. *Biometrika*, 40(3/4):237–264, 1953.

[126] Google Inc. 2-step verification: how it works. `www.google.com/accounts`, 2012.

[127] Google Inc. Manage your website passwords. `support.google.com/chrome`, 2012.

[128] F. T. Grampp and Robert Morris. UNIX operating system security. *AT&T Bell Labs Technical Journal*, 63(8):1649–1672, 1984.

[129] Virgil Griffith and Markus Jakobsson. Messin' with Texas: Deriving Mother's Maiden Names Using Public Records. *Applied Cryptography and Network Security*, 2005.

[130] Charles M. Grinstead and J. Laurie Snell. *Introduction to Probability.* American Mathematical Society, Providence, 2nd edition, 1997.

[131] The Trusted Computing Group. Trusted Computing Platform Alliance. Main Specification Version 1.1b., 2003. `www.trustedcomputinggroup.org/specs/TPM`.

[132] William J. Haga and Moshe Zviran. Question-and-Answer Passwords: An Empirical Evaluation. *Information Systems*, 16(3):335–343, 1991.

[133] J. Alex Halderman, Brent Waters, and Edward W. Felten. A Convenient Method for Securely Managing Passwords. In *WWW '05: Proceedings of the 14th International Conference on World Wide Web*, pages 471–479, New York, NY, USA, 2005. ACM.

[134] N. Haller, C. Metz, P. Nesser, and M. Straw. A One-Time Password System, 1998. RFC 2289.

[135] Neil Haller. The S/KEY One-Time Password System. In *NDSS '99: Proceedings of the 1999 Network and Distributed System Security Symposium*, 1994.

[136] Eran Hammer-Lahav and David Recordon. The OAuth 1.0 Protocol. `http://tools.ietf.org/html/draft-hammer-oauth-10`, 2010.

[137] Ralph V. Hartley. Transmission of Information. *Bell System Technical Journal*, 7(3):535–563, 1928.

[138] Martin Hellman. A Cryptanalytic Time-Memory Trade-Off. *IEEE Transactions on Information Theory*, 26(4):401–406, 1980.

[139] Cormac Herley. The Plight of the Targeted Attacker in a World of Scale. In *WEIS '10: Proceedings of the 9th Workshop on the Economics of Information Security*, 2010.

[140] Cormac Herley and Dinei Florêncio. Protecting Financial Institutions from Brute-Force Attacks. In *Proceedings of The IFIP TC 23rd International Information Security Conference*, pages 681–685, New York, NY, USA, 2008. Springer.

[141] Cormac Herley and Dinei Florêncio. Nobody Sells Gold for the Price of Silver: Dishonesty, Uncertainty and the Underground Economy. In *Economics of Information Security and Privacy*, pages 33–53. Springer US, 2010. 10.1007/978-1-4419-6967-5_3.

[142] Cormac Herley and P.C. van Oorschot. A Research Agenda Acknowledging the Persistence of Passwords. *IEEE Security & Privacy Magazine*, 2012.

[143] Cormac Herley, P.C. van Oorschot, and Andrew S. Patrick. Passwords: If We're So Smart, Why Are We Still Using Them? In *FC '09: The 13th International Conference on Financial Cryptography and Data Security*, Berlin, Heidelberg, 2009. Springer-Verlag.

[144] Nicholas J. Hopper and Manuel Blum. Secure Human Identification Protocols. In *ASIACRYPT '01: Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security*, pages 52–66, London, UK, 2001. Springer-Verlag.

[145] Facebook Inc. Facebook Developers: Authentication, 2012. `developers.facebook.com/docs/authentication/`.

[146] IronKey Inc. Protecting Commercial Online Banking Customers from Next-Generation Malware. www.ironkey.com, 2009.

[147] RSA Security Inc. RSA Mobile: two-factor authentication for a mobile world. 2002. `www.rsa.com`.

[148] RSA Security Inc. RSA SecurID Two-factor Authentication, 2010. `www.rsa.com`.

[149] XE Inc. Universal Currency Converter, 2012. `http://www.xe.com/ucc/`.

[150] Panagiotis G. Ipeirotis. Demographics of Mechanical Turk. Technical Report CEDER-10-01, New York University, 2010.

[151] Anthony Ivan and James Goodfellow. Improvements in or relating to Customer-Operated Dispensing Systems. UK Patent #GB1197183, 1966.

[152] Blake Ives, Kenneth R. Walsh, and Helmut Schneider. The Domino Effect of Password Reuse. *Communications of the ACM*, 47(4):75–78, 2004.

[153] Anil K. Jain, Arun Ross, and Sharath Pankanti. Biometrics: A Tool for Information Security. *IEEE Transactions on Information Forensics and Security*, 1(2):125–143, 2006.

[154] Markus Jakobsson and Ruj Akavipat. Rethinking Passwords to Adapt to Constrained Keyboards. `www.fastword.me`, 2011.

[155] Markus Jakobsson, Erik Stolterman, Susanne Wetzel, and Liu Yang. Love and Authentication. In *CHI '08: Proceedings of the 26th ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 197–200, New York, NY, USA, 2008. ACM.

[156] Markus Jakobsson, Liu Yang, and Susanne Wetzel. Quantifying the Security of Preference-Based Authentication. In *DIM '08: Proceedings of the 4th ACM Workshop on Digital Identity Management*, pages 61–70, New York, NY, USA, 2008. ACM.

[157] Ian Jermyn, Alain Mayer, Fabian Monrose, Michael K. Reiter, and Aviel D. Rubin. The Design and Analysis of Graphical Passwords. In *Proceedings of the 8th USENIX Security Symposium*, pages 1–14, Berkeley, CA, USA, 1999.

[158] Lei Jin, H. Takabi, and J.B.D. Joshi. Security and Privacy Risks of Using E-mail Address as an Identity. In *2nd International Conference on Social Computing*, pages 906–913, 2010.

[159] David L. Jobusch and Arthur E. Oldenhoeft. A Survey of Password Mechanisms: Weaknesses and Potential Improvements. Part 1. *Computers & Security*, 8:587–601, 1989.

[160] David L. Jobusch and Arthur E. Oldenhoeft. A Survey of Password Mechanisms: Weaknesses and Potential Improvements. Part 2. *Computers & Security*, 8:675–689, 1989.

[161] Arthur Evans Jr., William Kantrowitz, and Edwin Weiss. A User Authentication Scheme Not Requiring Secrecy in the Computer. *Communications of the ACM*, 17:437–442, 1974.

[162] Mike Just. Designing and Evaluating Challenge-Question Systems. *IEEE Security & Privacy*, 2(5):32–39, 2004.

[163] Mike Just and David Aspinall. Personal Choice and Challenge Questions: A Security and Usability Assessment. In *SOUPS '09: Proceedings of the 5th Symposium on Usable Privacy and Security*, 2009.

[164] Burt Kaliski. PKCS #5: Password-Based Cryptography Specification Version 2.0, 2000. RFC 2289.

[165] Chris Karlof, J. D. Tygar, and David Wagner. Conditioned-Safe Ceremonies and a User Study of an Application to Web Authentication. In *SOUPS '09: Proceedings of the 5th Symposium on Usable Privacy and Security*, New York, NY, USA, 2009. ACM.

[166] Jonathan Katz, Rafail Ostrovsky, and Moti Yung. Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords. In *EUROCRYPT '01: Proceedings of the 20th Annual International Conference on the Theory and Application of Cryptographic Techniques*, pages 475–494, London, UK, 2001. Springer-Verlag.

[167] Mark Keith, Benjamin Shao, and Paul John Steinbart. The usability of passphrases for authentication: An empirical field study. *International Journal of Human-Computer Studies*, 65(1):17–28, 2007.

[168] Patrick Gage Kelley, Saranga Komanduri, Michelle L. Mazurek, Rich Shay, Tim Vidas, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Julio Lopez. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. Technical Report CMU-CyLab-11-008, Carnegie Mellon University, 2011.

[169] Patrick Gage Kelley, Michelle L. Mazurek, Richard Shay, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, Saranga Komanduri, and Serge Egelman. Of Passwords and People: Measuring the Effect of Password-Composition Policies. In *CHI '11: Proceedings of the 29th ACM SIGCHI Conference on Human Factors in Computing Systems*, 2011.

[170] John Kelsey, Bruce Schneier, Chris Hall, and David Wagner. Secure Applications of Low-Entropy Keys. In *ISW '97: Proceedings of the 1st International Workshop on Information Security*, pages 121–134, London, UK, 1998. Springer-Verlag.

[171] Hyoungshick Kim, John Tang, and Ross Anderson. Social Authentication: Harder than it Looks. 2012.

[172] M.M. King. Rebus passwords. In *Proceedings of the 7th Annual Computer Security Applications Conference*, pages 239–243, 1991.

[173] Daniel Klein. Foiling the Cracker: A Survey of, and Improvements to, Password Security. In *Proceedings of the 2nd USENIX Security Workshop*, pages 5–14, 1990.

[174] Hugo Kleinhans, Jonathan Butts, and Sujeet Shenoi. Password Cracking Using Sony Playstations. In *Advances in Digital Forensics V*, volume 306 of *IFIP Advances in Information and Communication Technology*, pages 215–227. Springer Boston, 2009.

[175] Kazukuni Kobara and Hideki Imai. Limiting the Visible Space Visual Secret Sharing Schemes and Their Application to Human Identification. In *CRYPTO '96: Proceedings of the 16th Annual International Conference on Advances in Cryptology*, pages 185–195, London, UK, 1996. Springer-Verlag.

[176] J. Kohl and C. Neuman. The Kerberos Network Authentication Service (V5), 1993. RFC 1510.

[177] David P. Kormann and Aviel D. Rubin. Risks of the Passport single signon protocol. *Computer Networks*, 33(1-6):51–58, 2000.

[178] Markus Kuhn. Probability Theory for Pickpockets—ec-PIN Guessing. Technical report, Purdue University, 1997.

[179] Markus Kuhn. OTPW—a one-time password login package. `http://www.cl.cam.ac.uk/~mgk25/otpw.html`, 1998.

[180] S. Kullback and R. Leibler. On Information and Sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[181] Manu Kumar, Tal Garfinkel, Dan Boneh, and Terry Winograd. Reducing Shoulder-surfing by Using Gaze-based Password Entry. In *SOUPS '07: Proceedings of the 3rd Symposium on Usable Privacy and Security*, pages 13–19, New York, NY, USA, 2007. ACM.

[182] Sandeep Kumar, Christof Paar, Jan Pelzl, Gerd Pfeiffer, and Manfred Schimmler. Breaking Ciphers with COPACOBANA –A Cost-Optimized Parallel Code Breaker. In *CHES '06: Proceedings of 2006 Workshop on Cryptographic Hardware and Embedded Systems*, volume 4249, pages 101–118. Springer Berlin / Heidelberg, 2006.

[183] Cynthia Kuo, Sasha Romanosky, and Lorrie Faith Cranor. Human Selection of Mnemonic Phrase-based Passwords. In *SOUPS '06: Proceedings of the 2nd Symposium on Usable Privacy and Security*, pages 67–78. ACM, 2006.

[184] Leslie Lamport. Password authentication with insecure communication. *Communications of the ACM*, 24(11):770–772, 1981.

[185] LastPass. LastPass Security. `www.lastpass.com`, 2012.

[186] C. Latze and U. Ultes-Nitsche. Stronger Authentication in e-Commerce: How to Protect Even Naïve User Against Phishing, Pharming, and MITM Attacks. In *Proceedings of the IASTED International Conference on Communication Systems, Networks, and Applications*, pages 111–116, Anaheim, CA, USA, 2007. ACTA Press.

[187] Ben Laurie. OpenID: Phishing Heaven. `http://www.links.org/?p=187`, January 2007.

[188] Ben Laurie and Abe Singer. Choose the Red Pill and the Blue Pill. In *NSPW '08: Proceedings of the 2008 New Security Paradigms Workshop*, pages 127–133, New York, NY, USA, 2008. ACM.

[189] Sagi Leizerov. Privacy Advocacy Groups Versus Intel: A Case Study of How Social Movements Are Tactically Using the Internet to Fight Corporations. *Social Science Computer Review*, 18:461–483, 2000.

[190] Philip Leong and Chris Tham. UNIX Password Encryption Considered Insecure. In *Proceedings of the 2nd USENIX Security Workshop*, 1991.

[191] Ming Li and Paul Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer, 1997.

[192] Stan Z. Li and Anil K. Jain, editors. *Handbook of Face Recognition*. Springer, New York, NY, USA, 2005.

[193] Jack Lindamood and Murat Kantarcioglu. Inferring Private Information Using Social Network Data. Technical Report UTDCS-21-08, University of Texas at Dallas Computer Science Department, 2008.

[194] Cronto Ltd. Cronto's Visual Cryptogram. `www.cronto.com`, 2012.

[195] David J.C. Mackay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, 2003.

[196] David Malone and Wayne G. Sullivan. Guesswork and Entropy. In *Proceedings of the 2004 IEEE International Symposium on Information Theory*, volume 50, 2004.

[197] Udi Manber. A Simple Scheme to Make Passwords Based on One-Way Functions Much Harder To Crack. *Computers & Security*, 15(2):171–176, 1996.

[198] Mohammad Mannan and P.C. van Oorschot. Digital Objects as Passwords. In *HOTSEC'08: Proceedings of the 3rd Conference on Hot topics in Security*, pages 1–6, Berkeley, CA, USA, 2008. USENIX Association.

[199] Mohammad Mannan and P.C. van Oorschot. Leveraging Personal Devices for Stronger Password Authentication from Untrusted Computers. *Journal of Computer Security*, 19(4):703–750, 2011.

[200] Simon Marechal. Advances in password cracking. *Journal in Computer Virology*, 4:73–81, 2008.

[201] James L. Massey. Guessing and Entropy. In *Proceedings of the 1994 IEEE International Symposium on Information Theory*, page 204, 1994.

[202] T. Matsumoto, H. Matsumoto, K. Yamada, and S. Hoshino. Impact of Artificial "Gummy" Fingers on Fingerprint Systems. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 4677, pages 275–289, 2002.

[203] Tsutomu Matsumoto. Human-computer cryptography: An attempt. In *CCS '96: Proceedings of the 3rd ACM Conference on Computer and Communications Security*, pages 68–75, New York, NY, USA, 1996. ACM.

[204] Drew Mazurek. Central Authentication Service Protocol. `http://www.jasig.org/cas/protocol`, 2005.

[205] Jonathan M. McCune, Adrian Perrig, and Michael K. Reiter. Seeing-Is-Believing: Using Camera Phones for Human-Verifiable Authentication. In *SP '05: Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 110–124, Washington, DC, USA, 2005. IEEE Computer Society.

[206] R.J. McEliece and Zhong Yu. An inequality on entropy. In *Proceedings of the 1995 IEEE International Symposium on Information Theory*, 1995.

[207] Ari Medvinsky and Matthew Hur. Addition of Kerberos Cipher Suites to Transport Layer Security (TLS), 1999. RFC 2712.

[208] Andrew Mehler and Steven Skiena. Improving Usability Through Password-Corrective Hashing. In *String Processing and Information Retrieval*, volume 4209, pages 193–204. Springer Berlin / Heidelberg, 2006.

[209] Nele Mentens, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. Time-Memory Trade-Off Attack on FPGA Platforms: UNIX Password Cracking. In *Reconfigurable Computing: Architectures and Applications*, volume 3985. Springer Berlin / Heidelberg, 2006.

[210] Fabian Monrose and Aviel Rubin. Authentication via Keystroke Dynamics. In *CCS '97: Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 48–56, New York, NY, USA, 1997. ACM.

[211] R. L. "Bob" Morgan, Scott Cantor, Steven Carmody, Walter Hoehn, and Ken Klingenstein. Federated Security: The Shibboleth Approach. *EDUCAUSE Quarterly*, 27(4), 2004.

[212] Robert Morris and Ken Thompson. Password Security: A Case History. *Communications of the ACM*, 22(11):594–597, 1979.

[213] Alec Muffett. "Crack Version 4.1": A Sensible Password Checker for Unix. `http://www.crypticide.com/alecm/software/crack/crack-v4.1-whitepaper.pdf`, 1992.

[214] Steven Murdoch and Ross Anderson. Verified by Visa and MasterCard SecureCode: or, How Not to Design Authentication. In *FC '10: The 14th International Conference on Financial Cryptography and Data Security*. Springer-Verlag, 2010.

[215] Steven J. Murdoch. Hardened Stateless Session Cookies. In *16th International Workshop on Security Protocols*, pages 93–101, Berlin, Heidelberg, 2011. Springer-Verlag.

[216] Moni Naor and Benny Pinkas. Visual Authentication and Identification. In *CRYPTO '97: Proceedings of the 17th Annual International Conference on Advances in Cryptology*, volume 1294 of *Lecture Notes in Computer Science*, pages 322–336. Springer Berlin / Heidelberg, 1997.

[217] Moni Naor and Adi Shamir. Visual cryptography. In *EUROCRYPT '95: Proceedings of the 13th Annual International Conference on Theory and Applications of Cryptographic Techniques*, volume 950, pages 1–12. Springer Berlin / Heidelberg, 1995.

[218] Arvind Narayanan and Vitaly Shmatikov. Fast Dictionary Attacks on Passwords Using Time-Space Tradeoff. In *CCS '05: Proceedings of the 12th ACM Conference on Computer and Communications Security*, pages 364–372. ACM, 2005.

[219] Arvind Narayanan and Vitaly Shmatikov. How To Break Anonymity of the Netflix Prize Dataset. *eprint arXiv:cs/0610105*, 2006.

[220] Roger M. Needham and Michael D. Schroeder. Using Encryption for Authentication in Large Networks of Computers. *Communications of the ACM*, 21:993–999, 1978.

[221] Simon Nettle, Sean O'Neil, and Peter Lock. *PassWindow: A New Solution to Providing Second Factor Authentication*. VEST Corporation, 2009.

[222] Naom Nisan and Amnon Ta-Shma. Extracting Randomness: A Survey and New Constructions. *Journal of Computer and System Sciences*, 58(1):148–173, 1999.

[223] A. Nosseir, R. Connor, and M.D. Dunlop. Internet Authentication Based on Personal History—A Feasibility Test. In *Proceedings of the Customer Focused Mobile Services Workshop*. ACM Press, 2005.

[224] Gilbert Notoatmodjo and Clark Thomborson. Passwords and Perceptions. In Ljiljana Brankovic and Willy Susilo, editors, *AISC '09: The 7th Australasian Information Security Conference*, volume 98, pages 71–78, Wellington, New Zealand, 2009. ACS.

[225] Philippe Oechslin. Making a Faster Cryptanalytic Time-Memory Trade-Off. In *CRYPTO '03: Proceedings of the 23rd Annual International Conference on Advances in Cryptology*, 2003.

[226] U.S. Department of Labor Bureau of Labor Statistics. CPI Inflation Calculator, 2012. `http://www.bls.gov/data/inflation_calculator.htm`.

[227] L. O'Gorman. Comparing Passwords, Tokens, and Biometrics for User Authentication. In *Proceedings of the IEEE*, volume 91, pages 2021–2040, 2003.

[228] Rolf Oppliger. Microsoft .NET Passport: A Security Analysis. *Computer*, 36(7):29–35, 2003.

[229] Jim Owens and Jeanna Matthews. A Study of Passwords and Methods Used in Brute-Force SSH Attacks. Technical report, Clarkson University, 2008.

[230] Bryan Parno, Cynthia Kuo, and Adrian Perrig. Phoolproof Phishing Prevention. In *FC '06: The 10th International Conference on Financial Cryptography and Data Security*, volume 4107, pages 1–19. Springer Berlin / Heidelberg, 2006.

[231] Andreas Pashalidis and Chris J. Mitchell. *A Taxonomy of Single Sign-On Systems*, volume 2727 of *Information Security and Privacy*, pages 219–235. Springer, 2003.

[232] Andreas Pashalidis and Chris J. Mitchell. Impostor: A Single Sign-On System for Use from Untrusted Devices. In *Proceedings of IEEE Globecom*, 2004.

[233] Adrian Perrig and Dawn Song. Hash Visualization: a New Technique to Improve Real-World Security. In *In International Workshop on Cryptographic Techniques and E-Commerce*, pages 131–138, 1999.

[234] David Pesetsky. *Linguistic Universals and Universal Grammar*. The MIT Encyclopedia of the Cognitive Sciences. MIT Press, Cambridge, 1999.

[235] Benny Pinkas and Tomas Sander. Securing Passwords Against Dictionary Attacks. In *CCS '02: Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 161–170, New York, NY, USA, 2002. ACM.

[236] John O. Pliam. On the Incomparability of Entropy and Marginal Guesswork in Brute-Force Attacks. In *INDOCRYPT '00: The 1st International Conference on Cryptology in India*, 2000.

[237] Polybius. *Histories*. Perseus Project, Tufts University, 118 BCE. Accessed 2012.

[238] Rachael Pond, John Podd, Julie Bunnell, and Ron Henderson. Word Association Computer Passwords: The Effect of Formulation Techniques on Recall and Guessing Rates. *Computers & Security*, 19(7):645–656, 2000.

[239] Sigmund N. Porter. A Password Extension for Improved Human Factors. *Computers & Security*, 1(1):54–56, 1982.

[240] Brian Prince. Twitter Details Phishing Attacks Behind Password Reset. *eWeek*, 2010.

[241] Niels Provos and David Mazières. A Future-Adaptive Password Scheme. In *ATEC '99: Proceedings of the USENIX Annual Technical Conference*, pages 32–43, Berkeley, CA, USA, 1999. USENIX Association.

[242] George B. Purdy. A High Security Log-In Procedure. *Communications of the ACM*, 17:442–445, 1974.

[243] Ariel Rabkin. Personal knowledge questions for fallback authentication: Security questions in the era of Facebook. In *SOUPS '08: Proceedings of the 4th Symposium on Usable Privacy and Security*, pages 13–23, New York, NY, USA, 2008. ACM.

[244] Martin Rasmussen and Floyd W Rudmin. The coming PIN code epidemic: A survey study of memory of numeric security codes. *Electronic Journal of Applied Psychology*, 6(2):5–9, 2010.

[245] David Recordon and Dick Hardt. The OAuth 2.0 Protocol. `http://tools.ietf.org/html/draft-hammer-oauth2-00`, 2010.

[246] David Recordon and Drummond Reed. OpenID 2.0: a platform for user-centric identity management. In *DIM '06: Proceedings of the 2nd ACM Workshop on Digital Identity Management*, pages 11–16, New York, NY, USA, 2006. ACM.

[247] R.W. Reeder and S. Schechter. When the Password Doesn't Work: Secondary Authentication for Websites. *IEEE Security & Privacy Magazine*, 9(2):43–49, 2011.

[248] Arnold Reinhold. The Diceware Passphrase Project, 1995. `www.diceware.com`.

[249] Karen Renaud. Quantifying the Quality of Web Authentication Mechanisms: A Usability Perspective. *Journal of Web Engineering*, 3:95–123, 2004.

[250] Alfréd Rényi. On measures of information and entropy. In *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics and Probability*, pages 547–561, 1961.

[251] Bruce L. Riddle, Murray S. Miron, and Judith A. Semo. Passwords in use in a university timesharing environment. *Computers & Security*, 8(7):569–578, 1989.

[252] Shannon Riley. Password Security: What Users Know and What They Actually Do. *Usability News*, 8(1), 2006.

[253] Arun Ross, Jidnya Shah, and Anil K. Jain. From Template to Image: Reconstructing Fingerprints from Minutiae Points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):544–560, 2007.

[254] Blake Ross, Collin Jackson, Nick Miyake, Dan Boneh, and John C. Mitchell. Stronger Password Authentication Using Browser Extensions. In *Proceedings of the 14th USENIX Security Symposium*, pages 2–16, Berkeley, CA, USA, 2005. USENIX Association.

[255] Aviel D. Rubin. Independent One-Time Passwords. In *Proceedings of the 5th USENIX Security Symposium*, pages 15–24, Berkeley, CA, USA, 1995. USENIX Association.

[256] Nat Sakimura, John Bradley, Breno de Medeiros, Michael B. Jones, and Edmund Jay. OpenID Connect Standard 1.0 Draft 7, 2012. `http://openid.net/specs/openid-connect-standard-1_0.html`.

[257] Jerome H. Saltzer. Protection and the Control of Information Sharing in Multics. *Communications of the ACM*, 17:388–402, 1974.

[258] Hirokazu Sasamoto, Nicolas Christin, and Eiji Hayashi. Undercover: Authentication Usable in Front of Prying Eyes. In *CHI '08: Proceedings of the 26th ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 183–192, New York, NY, USA, 2008. ACM.

[259] Martina Angela Sasse, Sacha Brostoff, and Dirk Weirich. Transforming the 'Weakest Link'—a Human/Computer Interaction Approach to Usable and Effective Security. *BT Technology Journal*, 19(3):122–131, 2001.

[260] Darren Antwon Sawyer, William James Haga, and Moshe Zviran. The characteristics of user-generated passwords. Master's thesis, Naval Postgraduate School, Springfield, VA, USA, 1990.

[261] Stuart Schechter, A. J. Bernheim Brush, and Serge Egelman. It's No Secret: Measuring the security and reliability of authentication via 'secret' questions. In *SP '09: Proceedings of the 2009 IEEE Symposium on Security and Privacy*, pages 375–390, Washington, DC, USA, 2009. IEEE Computer Society.

[262] Stuart Schechter, Serge Egelman, and Robert W. Reeder. It's Not What You Know, But Who You Know: A social approach to last-resort authentication. In *CHI '09: Proceedings of the 27th ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 1983–1992, New York, NY, USA, 2009. ACM.

[263] Stuart Schechter, Cormac Herley, and Michael Mitzenmacher. Popularity is Everything: A new approach to protecting passwords from statistical-guessing attacks,. In *HotSec '10: The 5th USENIX Workshop on Hot Topics in Security*, 2010.

[264] Stuart Schechter and Robert W. Reeder. 1 + 1 = You: Measuring the comprehensibility of metaphors for configuring backup authentication. In *SOUPS '09: Proceedings of the 5th Symposium on Usable Privacy and Security*, pages 9:1–9:31, New York, NY, USA, 2009. ACM.

[265] Roland Schemers and Russ Allbery. WebAuth V3 Technical Specification. `http://webauth.stanford.edu/protocol.html`, 2009.

[266] Allan Lee Scherr. An analysis of time-shared computer systems. Technical report, Massachusetts Institute of Technology, 1965.

[267] Bruce Schneier. Real-World Passwords. *Schneier on Security*, December 2006.

[268] Donn Seeley. Password Cracking: A Game of Wits. *Communications of the ACM*, 32:700–703, 1989.

[269] Christian Seifert. Malicious SSH Login Attempts—Revisited. New Zealand Honeynet Alliance, 2006.

[270] Lee L. Selwyn. Computer Resource Accounting in a Time Sharing Environment. In *AFIPS '70 (Spring): Proceedings of the 1970 Spring Joint Computer Conference*, pages 119–130, New York, NY, USA, 1970. ACM.

[271] Claude E. Shannon. A Mathematical Theory of Communication. In *Bell System Technical Journal*, volume 7, pages 379–423, 1948.

[272] Claude E. Shannon. Prediction and entropy of printed English. In *Bell System Technical Journal*, volume 30, pages 50–64, 1951.

[273] Richard Shay, Saranga Komanduri, Patrick Gage Kelley, Pedro Giovanni Leon, Michelle L. Mazurek, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Encountering Stronger Password Requirements: User Attitudes and Behaviors. In *SOUPS '10: Proceedings of the 6th Symposium on Usable Privacy and Security*. ACM, 2010.

[274] Herbert Sichel. On a Distribution Law for Word Frequencies. *Journal of the American Statistical Association*, 1975.

[275] Kamaljit Singh. On Improvements to Password Security. *SIGOPS Operating Systems Review*, 19:53–60, 1985.

[276] Supriya Singh, Anuja Cabraal, Catherine Demosthenous, Gunela Astbrink, and Michele Furlong. Password Sharing: Implications for Security Design Based on Social Practice. In *CHI '07: Proceedings of the 25th ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 895–904, New York, NY, USA, 2007. ACM.

[277] Sidney L. Smith. Authenticating Users by Word Associations. *Computers & Security*, 6:464–470, 1987.

[278] Eugene Spafford. The Internet Worm Program: An Analysis. *SIGCOMM Computers and Communications Review*, 19:17–57, 1989.

[279] Eugene Spafford. Observations on Reusable Password Choices. In *Proceedings of the 3rd USENIX Security Workshop*, 1992.

[280] Yishay Spector and Jacob Ginzberg. Pass-sentence—a new approach to computer code. *Computers & Security*, 13(2):145–160, 1994.

[281] Martijn Sprengers. GPU-based Password Cracking. Master's thesis, Radboud University Nijmegen, 2011.

[282] Frank Stajano. Pico: No more passwords! In *19th International Workshop on Security Protocols*, volume 7114. Springer, 2011.

[283] Jennifer G. Steiner, Clifford Neuman, and Jeffrey I. Schiller. Kerberos: An Authentication Service for Open Network Systems. In *USENIX Winter Conference*, pages 191–202, 1988.

[284] Brett Stone-Gross, Marco Cova, Lorenzo Cavallaro, Bob Gilbert, Martin Szydlowski, Richard Kemmerer, Christopher Kruegel, and Giovanni Vigna. Your botnet is my botnet: Analysis of a botnet takeover. In *CCS '09: Proceedings of the 16th ACM Conference on Computer and Communications Security*, pages 635–647. ACM, 2009.

[285] Adam Stubblefield and Dan Simon. Inkblot Authentication. Technical Report MSR-TR-2004-85, Microsoft Research, 2004.

[286] San-Tsai Sun, Yazan Boshmaf, Kirstie Hawkey, and Konstantin Beznosov. A Billion Keys, but Few Locks: The Crisis of Web Single Sign-On. In *NSPW '10: Proceedings of the 2010 New Security Paradigms Workshop*, pages 61–72, New York, NY, USA, 2010. ACM.

[287] San-Tsai Sun, Kirstie Hawkey, and Konstantin Beznosov. $\text{OpenID}_{\text{email}}$ Enabled Browser: Towards Fixing the Broken Web Single Sign-On Triangle. In *DIM '10: Proceedings of the 6th ACM workshop on Digital Identity Management*, pages 49–58, New York, NY, USA, 2010. ACM.

[288] Latanya Sweeney. Uniqueness of Simple Demographics in the U.S. Population. Technical Report LIDAP-WP4, Carnegie Mellon University, 2000.

[289] Latanya Sweeney. k-Anonymity: A model for protecting privacy. *International Journal on Uncertainty and Fuzziness in Knowledge-Based Systems*, 10:557–570, 2002.

[290] H. Tao. Pass-Go, a new graphical password scheme. Master's thesis, Carleton University, 2006.

[291] Furkan Tari, A. Ant Ozok, and Stephen H. Holden. A Comparison of Perceived and Real Shoulder-Surfing Risks Between Alphanumeric and Graphical Passwords. In *SOUPS '06: Proceedings of the 2nd Symposium on Usable Privacy and Security*, pages 56–66, New York, NY, USA, 2006. ACM.

[292] David Taylor, Thomas Wu, and Trevor Perrin. Using the Secure Remote Password (SRP) Protocol for TLS Authentication, 2007. RFC 5054.

[293] Henri Theil. *Economic Forecasts and Policy*. North, Amsterdam, 1961.

[294] Julie Thorpe and P.C. van Oorschot. Human-Seeded Attacks and Exploiting Hot-Spots in Graphical Passwords. In *Proceedings of 16th USENIX Security Symposium*, Berkeley, CA, USA, 2007.

[295] Michael Toomim, Xianhang Zhang, James Fogarty, and James A. Landay. Access Control by Testing for Shared Knowledge. In *CHI '08: Proceedings of the 26th ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 193–196, 2008.

[296] Antoine Galland (translator). *The Arabian Nights Entertainments*, volume 17. Harrison and Co., London, 1785.

[297] Tim Valentine. An Evaluation of the Passfaces Personal Authentication System. Technical report, Goldmsiths College University of London, 1998.

[298] Gregory Valiant and Paul Valiant. Estimating the unseen: A sublinear-sample canonical estimator of distributions. In *STOC '11: Proceedings of the 43rd Symposium on Theory of Computing*, 2011.

[299] Paul Valiant. *Testing Symmetric Properties of Distributions*. PhD thesis, Massachusetts Institute of Technology, 2008.

[300] Timothy W. van der Horst and Kent E. Seamons. Simple Authentication for the Web. In *3rd International Conference on Security and Privacy in Communications Networks and the Workshops, 2007*, pages 473–482, 2007.

[301] P.C. van Oorschot and Julie Thorpe. On Predictive Models and User-Drawn Graphical Passwords. *ACM Transactions on Information Systems Security*, 10(4):1–33, 2008.

[302] David Wagner and Ian Goldberg. Proofs of Security for the Unix Password Hashing Algorithm. In *ASIACRYPT '00: Proceedings of the 6th International Conference on the Theory and Application of Cryptology and Information Security*, volume 1976, pages 560–572. Springer Berlin / Heidelberg, 2000.

[303] David Walden and Tom Van Vleck, editors. *The Compatible Time Sharing System (1961–1973) Fiftieth Anniversary Commemorative Overview*. Washington: IEEE Computer Society.

[304] Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In *EUROCRYPT '05: Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques*, pages 19–35, Berlin, Heidelberg, 2005. Springer-Verlag.

[305] Jon Warbrick. The Cambridge Web Authentication System: WAA->WLS communication protocol. `http://raven.cam.ac.uk/project/waa2wls-protocol.txt`, 2005.

[306] Rick Wash. Folk Models of Home Computer Security. In *SOUPS '10: Proceedings of the 6th Symposium on Usable Privacy and Security*, New York, NY, USA, 2010. ACM.

[307] Richard Weber. The Statistical Security of GrIDsure. `www.gridsure.com`, 2006.

[308] Daphna Weinshall. Cognitive Authentication Schemes Safe Against Spyware (Short Paper). In *SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 295–300, Washington, DC, USA, 2006. IEEE Computer Society.

[309] Matt Weir, Sudhir Aggarwal, Michael Collins, and Henry Stern. Testing Metrics for Password Creation Policies by Attacking Large Sets of Revealed Passwords. In *CCS '10: Proceedings of the 17th ACM Conference on Computer and Communications Security*, pages 162–175. ACM, 2010.

[310] Matt Weir, Sudhir Aggarwal, Breno de Medeiros, and Bill Glodek. Password Cracking Using Probabilistic Context-Free Grammars. In *SP '09: Proceedings of the 2009 IEEE Symposium on Security and Privacy*, pages 391–405. IEEE, 2009.

[311] Dirk Weirich and Martina Angela Sasse. Pretty Good Persuasion: A First Step Towards Effective Password Security in the Real World. In *NSPW '01: Proceedings of the 2001 on New Security Paradigms Workshop*, pages 137–143, New York, NY, USA, 2001. ACM.

[312] Susan Wiedenbeck, Jim Waters, Jean-Camille Birget, Alex Brodskiy, and Nasir Memon. PassPoints: Design and longitudinal evaluation of a graphical password system. *International Journal of Human-Computer Studies*, 63:102–127, 2005.

[313] Alexander Wiesmaier, Marcus Fischer, Evangelos G. Karatsiolis, and Marcus Lippert. Outflanking and Securely Using the PIN/TAN-System. *Computing Research Repository ePrint*, cs.CR/0410025, 2004.

[314] Maurice V. Wilkes. *Time-sharing computer systems*. Elsevier, New York, 1968.

[315] Hugh Wimberly and Lorie M. Liebrock. Using Fingerprint Authentication to Reduce System Security: An Empirical Study. In *SP '11: Proceedings of the 2011 IEEE Symposium on Security and Privacy*, pages 32–46, Washington, DC, USA, 2011. IEEE Computer Society.

[316] Ford-Long Wong and Frank Stajano. Multi-channel protocols. In *13th International Workshop on Security Protocols*, pages 112–127, Berlin, Heidelberg, 2005. Springer-Verlag.

[317] Min Wu, Simson Garfinkel, and Rob Miller. Secure Web Authentication with Mobile Phones. In *DIMACS Workshop on Usable Privacy and Security Software*, 2004.

[318] Thomas Wu. The Secure Remote Password Protocol. In *NDSS '98: Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium*, 1998.

[319] Thomas Wu. A Real-World Analysis of Kerberos Password Security. In *NDSS '99: Proceedings of the 1999 Network and Distributed System Security Symposium*, 1999.

[320] Roman V. Yampolskiy and Venu Govindaraju. Behavioural biometrics: a survey and classification. *International Journal of Biometrics*, 1:81–113, 2008.

[321] Jeff Yan. A Note on Proactive Password Checking. In *NSPW '01: Proceedings of the 2001 New Security Paradigms Workshop*, pages 127–135, New York, NY, USA, 2001. ACM.

[322] Jeff Yan, Alan Blackwell, Ross Anderson, and Alasdair Grant. Password Memorability and Security: Empirical Results. *IEEE Security & Privacy Magazine*, 2(5):25–34, 2004.

[323] Qiang Yan, Jin Han, Yingjiu Li, and Robert Huijie Deng. On Limitations of Designing Usable Leakage-Resilient Password Systems: Attacks, Principles and Usability. In *NDSS '12: Proceedings of the Network and Distributed System Security Symposium*, 2012.

[324] Sarita Yardi, Nick Feamster, and Amy Bruckman. Photo-Based Authentication Using Social Networks. In *WOSN '08: Proceedings of the 1st Workshop on Online Social Networks*, pages 55–60, New York, NY, USA, 2008. ACM.

[325] Yubico. YubiKey Security Overview. `www.yubico.com`.

[326] Yinqian Zhang, Fabian Monrose, and Michael K. Reiter. The Security of Modern Password Expiration: An Algorithmic Framework and Empirical Analysis. In *CCS '10: Proceedings of the 17th ACM Conference on Computer and Communications Security*, pages 176–186. ACM, 2010.

[327] Philip R. Zimmermann. *The Official PGP User's Guide*. MIT Press, 1995.

[328] Moshe Zviran and William J. Haga. A Comparison of Password Techniques for Multilevel Authentication Mechanisms. *Computer Journal*, 36(3):227–237, 1993.

Appendix A

Glossary of symbols

The following is a summary of all notation used throughout the text:

symbol	meaning	introduced
α	the proportion of accounts an attacker is seeking to compromise in a guessing attack	§3.2.3
α_*	the estimated upper limit for which naive estimates for $\tilde{\mu}_\alpha$ and $\tilde{G}_\alpha$ will be accurate	§5.5
α_m	the cumulative probability of all events which occurred at least m times in a sample	§5.5
β	the number of attempts per account an attacker is willing to make in a guessing attack	§3.2.1
λ_β	β-success-rate, the cumulative probability of success after β guesses	§3.2.1
μ_α	α-work-factor, the minimum number of events with cumulative probability α	§3.2.2
a	the scaling exponent in a power-law distribution	5.6
$\mathbf{c}$	a challenge sent by the verifier during an authentication protocol	§1.1.1
d	a demographic predicate function evaluated during an anonymised password collection experiment	§6.1.4
f	the observed frequency of an event in a sample	§5
f_x	the observed frequency of event x in a sample	§5
f_i	the observed frequency of the i^{th} most frequent event in a sample	§5
f_i^{GT}	the Good-Turing adjusted frequency of the i^{th} most frequent event in a sample	§5.4

symbol	meaning	introduced
f_i^{SGT}	the Simple Good-Turing adjusted frequency of the i^{th} most frequent event in a sample	§5.4
$\mathbf{i}$	a user's identity transmitted in an authentication protocol	§1.1.1
i	the index of an item in a distribution, ranked by decreasing probability	§1.4
j	the index of an item in an approximate distribution, when performing a sub-optimal guessing attack	§8.1
k	the minimum size of a group for which results are reported in an anonymised data collection experiment	§6.1.4
m	the number of times an event was observed in a sample	§5
m_*	the estimated lower limit for the number of times an event was observed in a sample which can be used in naive estimates for $\tilde{\mu}_\alpha$ and $\tilde{G}_\alpha$ which will be accurate	§5.5
p	the probability of an individual event in a distribution	§1.4
p_x	the probability of event x in a distribution	§1.4
p_i	the probability of the i^{th} most probable event in a distribution	§1.4
p_i^{GT}	the Good-Turing estimated probability of the i^{th} most frequent event in a sample	§5.4
p_i^{SGT}	the Simple Good-Turing estimated probability of the i^{th} most frequent event in a sample	§5.4
r	a strong random nonce used to anonymise collected data	§6.1.2
s	the scaling exponent in a Zipf distribution	§9.2.2
$\mathbf{x}$	an individual password transmitted in an authentication protocol	§1.1.1
x	an individual event in a distribution	§1.4
x_i	the i^{th} most probable event in a distribution	§1.4
G	guesswork or guessing entropy	§3.1.3
G_α	α-guesswork (guesswork for an attacker with desired success rate α)	§3.2.3
$\mathbf{H}$	a cryptographic hash function used to anonymise data	§6.1.4
$\mathcal{H}$	a histogram used in a password collection experiment	§3.1.2
H_n	Rényi entropy of order n	§3.1.2
H_0	Hartley entropy	§3.1.2
H_1	Shannon entropy	§3.1.1
H_2	collision entropy	§3.1.2
H_∞	min-entropy	§3.1.2
M	the total number of observations in a sample distribution	§5

symbol	meaning	introduced
N	the total number of items in a discrete probability distribution	§1.4
$\mathcal{P}$	an arbitrary property of a distribution	8.1.2
S^{G}	the partial guessing strength metric for a single event	§9.1.3
S^{I}	the index strength metric for a single event	§9.1.2
S^{NIST}	the NIST entropy estimation formula [58] for an individual password	§9.3
S^{P}	the probability strength metric for a single event	§9.1.1
$V(M)$	the number of distinct items observed in a sample of size M	§5
$V(m, M)$	the number of distinct items observed exactly m times each in a sample of size M	§5
$V^*(m, M)$	the ideal number of distinct items observed exactly m times each in a sample of size M which would make all naive estimates accurate	§5.5
$V^{\mathrm{bs}}(m, M)$	the average number of distinct items observed exactly m times in a bootstrap resample from a sample of size M	§5.5
$X \xleftarrow{\mathrm{R}} \mathcal{X}$	a random variable (unknown event) drawn at random from $\mathcal{X}$	§1.4
$\mathcal{U}_N$	a discrete uniform distribution with N equiprobable events	§1.4
$\mathcal{X}$	a discrete probability distribution	§1.4
$\hat{\mathcal{X}}$	an empirical distribution of random samples from the distribution $\mathcal{X}$	§5.1
$\mathcal{X} \parallel \mathcal{Y}$	a partial guessing attack on $\mathcal{X}$ using $\mathcal{Y}$ as an approximate distribution to choose guesses	§8.1.1
$\mathcal{Y}$	the approximate distribution used to perform a sub-optimal guessing attack on a separate distribution $\mathcal{X}$	§1.4
Z	the Zipfian approximation for frequency counts used during Simple Good-Turing approximation	§5.4
	modifiers	
$\tilde{\mathcal{P}}$	a bit-converted equivalent (effective key-length) of the property $\mathcal{P}$	§3.2.4
$\hat{\mathcal{P}}$	a naive estimate of the property $\mathcal{P}$ based on a randomly sampled distribution	§5.1
$\bar{\mathcal{P}}$	the average value of $\mathcal{P}$ computed across many random subsamples	§5.5
$D_{\mathcal{P}}$	the divergence, or difference in the value of $\mathcal{P}$ in a sub-optimal attack compared to an optimal one	§8.1.1

Appendix B

Additional proofs of theorems

B.1 Lower bound on G_1 for mixture distributions

We prove a lower bound on G_1 for mixture distributions as claimed in §3.3.5:

Theorem B.1.1. *For a mixture distribution $\mathcal{Z} = q_{\mathcal{X}} \cdot \mathcal{X} + q_{\mathcal{Y}} \cdot \mathcal{Y}$, we have $G_1(\mathcal{Z}) \geq q_{\mathcal{X}} \cdot G_1(\mathcal{X}) + q_{\mathcal{Y}} \cdot G_1(\mathcal{Y})$.*

Proof. We can prove this result by reduction to a simpler problem. Suppose that an adversary must guess a value $Z \xleftarrow{\mathrm{R}} \mathcal{Z}$ but she will be told if Z was actually drawn from $\mathcal{X}$ or $\mathcal{Y}$. By definition, she will require $G_1(\mathcal{X})$ guesses in the first case and $G_1(\mathcal{Y})$ in the second. Because expectation is linear, this means she requires an expected $q_{\mathcal{X}} \cdot G_1(\mathcal{X}) + q_{\mathcal{Y}} \cdot G_1(\mathcal{Y})$ guesses for random Z. An adversary who is *not* told which underlying distribution Z was drawn from has strictly less information and hence can do no better than this limit, proving the result. □

B.2 Bounds between $\tilde{G}_\alpha$ and $\tilde{\mu}_\alpha$

We prove the close bounds between $\tilde{G}_\alpha$ and $\tilde{\mu}_\alpha$ claimed in Table 3.1.

Theorem B.2.1. *For any $\mathcal{X}, \alpha$, it holds that $\tilde{\mu}_\alpha(\mathcal{X}) + \lg(1-\alpha) \leq \tilde{G}_\alpha(\mathcal{X}) \leq \tilde{\mu}_\alpha(\mathcal{X})$.*

Proof. To prove this bound, we first re-write the definition of G_α from Equation 3.10:

$$\begin{aligned} G_\alpha(\mathcal{X}) &= (1 - \llbracket\alpha\rrbracket) \cdot \mu_\alpha(\mathcal{X}) + \sum_{i=1}^{\mu_\alpha(\mathcal{X})} p_i \cdot i \\ &= (1 - \llbracket\alpha\rrbracket) \cdot \mu_\alpha(\mathcal{X}) + \sum_{i=1}^{\mu_\alpha(\mathcal{X})} p_i \cdot \mu_\alpha(\mathcal{X}) - p_i \cdot (\mu_\alpha(\mathcal{X}) - i) \\ &= \mu_\alpha(\mathcal{X}) - \sum_{i=1}^{\mu_\alpha(\mathcal{X})} p_i \cdot (\mu_\alpha(\mathcal{X}) - i) \end{aligned}$$

The term $\sum_{i=1}^{\mu_\alpha(\mathcal{X})} p_i \cdot (\mu_\alpha(\mathcal{X}) - i)$, which represents the difference between $\tilde{G}_\alpha$ and $\tilde{\mu}_\alpha$, will be maximised as $p_1 \to \alpha$ when it is $\alpha \cdot (\mu_\alpha(\mathcal{X}) - 1)$. It will take on its minimal value in the case when the benefit of stopping early is lowest, which occurs if $\mathcal{X}$ has uniform probability over the first μ_α elements. In this case, it will be:

$$\begin{aligned} \sum_{i=1}^{\mu_\alpha(\mathcal{X})} p_i \cdot (\mu_\alpha(\mathcal{X}) - i) &= \sum_{i=1}^{\mu_\alpha(\mathcal{X})} \frac{\llbracket\alpha\rrbracket}{\mu_\alpha(\mathcal{X})} \cdot (\mu_\alpha(\mathcal{X}) - i) \\ &= \frac{\llbracket\alpha\rrbracket}{\mu_\alpha(\mathcal{X})} \cdot \sum_{i=1}^{\mu_\alpha(\mathcal{X})} (\mu_\alpha(\mathcal{X}) - i) \\ &= \frac{\llbracket\alpha\rrbracket}{\mu_\alpha(\mathcal{X})} \cdot \frac{\mu_\alpha(\mathcal{X})\,(\mu_\alpha(\mathcal{X}) - 1)}{2} \\ &= \frac{\llbracket\alpha\rrbracket}{2} \cdot (\mu_\alpha(\mathcal{X}) - 1) \end{aligned}$$

Note that the upper and lower bound case differ only by a factor of 2, we can generalise to say that for some $\frac{1}{2} \leq \gamma \leq 1$, we have:

$$\begin{aligned} G_\alpha(\mathcal{X}) &= \mu_\alpha(\mathcal{X}) - \gamma\llbracket\alpha\rrbracket \cdot (\mu_\alpha(\mathcal{X}) - 1) \\ &= (1 - \gamma\llbracket\alpha\rrbracket)\mu_\alpha(\mathcal{X}) + \gamma\llbracket\alpha\rrbracket \end{aligned}$$

Now we can plug this value into the definition of $\tilde{G}_\alpha$ from Equation 3.16:

$$\begin{aligned} \tilde{G}_\alpha(\mathcal{X}) &= \lg\left[\frac{2 \cdot G_\alpha(\mathcal{X})}{\llbracket\alpha\rrbracket} - 1\right] - \lg(2 - \llbracket\alpha\rrbracket) \\ &= \lg\left[\frac{2 \cdot ((1 - \gamma\llbracket\alpha\rrbracket)\mu_\alpha(\mathcal{X}) + \gamma\llbracket\alpha\rrbracket)}{\llbracket\alpha\rrbracket} - 1\right] - \lg(2 - \llbracket\alpha\rrbracket) \\ &= \lg\left[\frac{2 \cdot (1 - \gamma\llbracket\alpha\rrbracket)\mu_\alpha(\mathcal{X})}{\llbracket\alpha\rrbracket} - 1 + 2\gamma\right] - \lg(2 - \llbracket\alpha\rrbracket) \end{aligned}$$

Returning to our limit that $\frac{1}{2} \leq \gamma \leq 1$, we now have bounds of:

$$\begin{aligned}
\lg\left[\frac{2\cdot(1-\lceil\alpha\rceil)\mu_\alpha(\mathcal{X})}{\lceil\alpha\rceil}+1\right]-\lg(2-\lceil\alpha\rceil) \leq\ & \tilde{G}_\alpha(\mathcal{X}) \leq \lg\left[\frac{2\cdot(1-\frac{\lceil\alpha\rceil}{2})\mu_\alpha(\mathcal{X})}{\lceil\alpha\rceil}\right]-\lg(2-\lceil\alpha\rceil)\\
\lg\left[\frac{2\cdot(1-\lceil\alpha\rceil)\mu_\alpha(\mathcal{X})}{\lceil\alpha\rceil}\right]-\lg(2) \leq\ & \tilde{G}_\alpha(\mathcal{X}) \leq \lg\left[\frac{(2-\lceil\alpha\rceil)\mu_\alpha(\mathcal{X})}{\lceil\alpha\rceil}\right]-\lg(2-\lceil\alpha\rceil)\\
\lg\left[\frac{\mu_\alpha(\mathcal{X})}{\lceil\alpha\rceil}\right]+\lg(1-\lceil\alpha\rceil) \leq\ & \tilde{G}_\alpha(\mathcal{X}) \leq \lg\left[\frac{\mu_\alpha(\mathcal{X})}{\lceil\alpha\rceil}\right]\\
\tilde{\mu}_\alpha(\mathcal{X})+\lg(1-\lceil\alpha\rceil) \leq\ & \tilde{G}_\alpha(\mathcal{X}) \leq \tilde{\mu}_\alpha(\mathcal{X})
\end{aligned}$$

□

B.3 Non-comparability of $\tilde{\lambda}_\beta$ with $\tilde{G}_1$ and H_1

Theorem B.3.1. *Given any $\delta > 0$, $\beta > 0$, there exists a distribution $\mathcal{X}$ such that $\tilde{\lambda}_\beta(\mathcal{X}) < H_1(\mathcal{X}) - \delta$ and $\tilde{\lambda}_\beta(\mathcal{X}) < \tilde{G}_1(\mathcal{X}) - \delta$.*

Proof. We prove this result, similar to Theorem 3.3.1, by constructing a pathological distribution with constant $\tilde{\lambda}_\beta$ and $H_1 \geq \tilde{\lambda}_\beta + \delta + 1$. The distribution must then also have $\tilde{G} \geq \tilde{\lambda}_\beta + \delta$ by Massey's bound (§3.3.2).

We'll create a mixture distribution $\mathcal{X} = \frac{1}{2}\cdot\mathcal{U}_\beta + \frac{1}{2}\cdot\mathcal{U}_\gamma$. That is, $\mathcal{X}$ has β common items and γ uncommon ones.[1] The β-success-rate is trivially $\lambda_\beta = \frac{1}{2}$. The converted β-success-rate will then be:

$$\begin{aligned}
\tilde{\lambda}_\beta(\mathcal{X}) &= \lg\left(\frac{\beta}{\lambda_\beta(\mathcal{X})}\right)\\
&= \lg\left(\frac{\beta}{\frac{1}{2}}\right)\\
&= \lg\beta + 1
\end{aligned}$$

The Shannon entropy will be:

[1] This proof can be done with tighter bounds by selecting from $\mathcal{U}_\beta$ with probablity $> \frac{1}{2}$, but we choose the algebraically simpler approach here.

$$\begin{aligned} H_1(\mathcal{X}) &= \sum_{i=1}^{N} -p_i \lg p_i \\ &= \beta \cdot \frac{1}{2\beta} \cdot -\lg \frac{1}{2\beta} + \gamma \cdot \frac{1}{2\gamma} \cdot -\lg \frac{1}{2\gamma} \\ &= \frac{1}{2} \cdot (\lg \beta + 1) + \frac{1}{2} \cdot (\lg \gamma + 1) \\ &= \frac{\lg \beta}{2} + \frac{\lg \gamma}{2} + 1 \end{aligned}$$

To complete the proof we solve for γ to ensure that our separation δ is met:

$$\begin{aligned} H_1(\mathcal{X}) &\geq \tilde{\lambda}_\beta(\mathcal{X}) + \delta + 1 \\ \frac{\lg \beta}{2} + \frac{\lg \gamma}{2} + 1 &\geq \lg \beta + 1 + \delta + 1 \\ \frac{\lg \gamma}{2} &\geq \frac{\lg \beta}{2} + \delta + 1 \\ \lg \gamma &\geq \lg \beta + 2\delta + 2 \\ \gamma &\geq 2^{\lg \beta + 2\delta + 2} \\ \gamma &\geq \beta \cdot 4^{\delta+1} \end{aligned}$$

This completes the proof. □

Note that once again we are left with an exponential size requirement that $|\mathcal{X}| \in \Theta\left(\beta \cdot 4^{\delta}\right)$.

B.4 Non-additivity of partial guessing metrics

We prove the non-additivity of $\tilde{\lambda}_\beta$, $\tilde{\mu}_\alpha$ and $\tilde{G}_\alpha$, as claimed in §3.3.5, in three separate theorems:

Theorem B.4.1. *For any $k \geq 1$, $\beta \geq 1$ and $\varepsilon > 0$, there exists a sequence of distributions $\mathcal{X}_1, \ldots, \mathcal{X}_k$ such that $\tilde{\lambda}_\beta(\mathcal{X}_1, \ldots, \mathcal{X}_k) \leq \max\limits_{1 \leq i \leq k} \left\{ \tilde{\lambda}_\beta(\mathcal{X}_i) \right\} + \varepsilon$.*

Proof. Let all the distributions be identical, $\forall_i \mathcal{X}_i = \mathcal{X}$, and let $\mathcal{X}$ contain one event with very high probability $1 - \upsilon$ and an arbitrary number of events of probability $\leq \upsilon$. We can see that λ_1, the probability of success with one guess, is $\lambda_1(\mathcal{X}) = 1 - \upsilon$ and thus $\lambda_\beta(\mathcal{X}) \geq 1 - \upsilon$ for any $\beta \geq 1$. When guessing k variables λ_1 is equal to the probability that the most likely event occurs for all, which is $\lambda_1(\mathcal{X}^k) = (1 - \upsilon)^k$, and again $\lambda_\beta(\mathcal{X}^k) \geq (1 - \upsilon)^k$ for any $\beta \geq 1$. Given

that $\tilde{\lambda}_\beta(\mathcal{X}) \geq \lg\left(\frac{\beta}{1-\upsilon}\right)$ and $\tilde{\lambda}_\beta(\mathcal{X}^k) \geq \lg\left(\frac{\beta}{(1-\upsilon)^k}\right)$, we can find the required value of υ for a given ε:

$$\begin{aligned}
\lg\left(\frac{\beta}{(1-\upsilon)^k}\right) &\leq \lg\left(\frac{\beta}{1-\upsilon}\right) + \varepsilon \\
\lg\beta - k\cdot\lg(1-\upsilon) &\leq \lg\beta - \lg(1-\upsilon) + \varepsilon \\
\lg(1-\upsilon) &\geq \frac{\varepsilon}{1-k} \\
\upsilon &\geq 1 - 2^{\frac{\varepsilon}{1-k}}
\end{aligned}$$

□

Theorem B.4.2. *For any $k \geq 1$, $0 < \alpha < 1$ and $\varepsilon > 0$, there exists a sequence of distributions $\mathcal{X}_1, \ldots, \mathcal{X}_k$ such that $\tilde{\mu}_\alpha(\mathcal{X}_1, \ldots, \mathcal{X}_m) \leq \max\limits_{1\leq i\leq k}\{\tilde{\mu}_\alpha(\mathcal{X}_i)\} + \varepsilon$.*

Proof. Again we choose all the distributions to be identical, $\forall_i \mathcal{X}_i = \mathcal{X}$, with $\mathcal{X}$ containing one event of probability $1-\upsilon$ and an arbitrary number of events of probability $\leq \upsilon$. We can choose $1-\upsilon \geq \alpha$ which sets $\mu_\alpha = 1$, because the first guess will have probability $\geq \alpha$. The probability of guessing all k variables correctly in one guess is $(1-\upsilon)^k$, so we'll set $(1-\upsilon)^k \geq \alpha$ to give us $\mu_\alpha(\mathcal{X}) = \mu_\alpha(\mathcal{X}^k) = 1$. Solving for this, we get that $\upsilon \leq 1 - \sqrt[k]{\alpha}$.

Recalling that $\tilde{\mu}_\alpha(\mathcal{X}) = \lg\left(\frac{\mu_\alpha(\mathcal{X})}{\lceil\!\lceil\alpha\rceil\!\rceil}\right)$, we have $\mu_\alpha(\mathcal{X}) = \mu_\alpha(\mathcal{X}^k) = 1$ by choice and $\lceil\!\lceil\alpha\rceil\!\rceil = 1-\upsilon$ and $(1-\upsilon)^k$ for $\mathcal{X}$ and $\mathcal{X}^k$, respectively. Therefore, we need to satisfy:

$$\begin{aligned}
\lg\left(\frac{1}{(1-\upsilon)^k}\right) &\leq \lg\left(\frac{1}{1-\upsilon}\right) + \varepsilon \\
-k\lg(1-\upsilon) &\leq -\lg(1-\upsilon) + \varepsilon \\
(1-k)\lg(1-\upsilon) &\leq \varepsilon \\
\lg(1-\upsilon) &\geq \frac{\varepsilon}{1-k} \\
\upsilon &\geq 1 - 2^{\frac{\varepsilon}{1-k}}
\end{aligned}$$

Interestingly, this is the same condition as in the proof for Theorem B.4.1 despite the slightly more complicated formula. □

Theorem B.4.3. *For any $k \geq 1$, $0 < \alpha < 1$ and $\varepsilon > 0$, there exists a sequence of distributions $\mathcal{X}_1, \ldots, \mathcal{X}_k$ such that $\tilde{G}_\alpha(\mathcal{X}_1, \ldots, \mathcal{X}_k) \leq \max\limits_{1\leq i\leq k}\left\{\tilde{G}_\alpha(\mathcal{X}_i)\right\} + \varepsilon$.*

Proof. By the definition of G_α in Equation 3.10, for $\alpha \leq p_1$ it will hold that $G_\alpha = \mu_\alpha$. Therefore this theorem can be proved identically to the proof of Theorem B.4.2 above, which constructed a distribution with one event of probability $p_1 > \alpha$. □

B.5 Expected value of index strength metric $S^{\mathbf{I}}(x)$ for a uniform distribution

As claimed in §9.1.2, using the definition from Equation 9.5 that $S^{\mathrm{I}}_{\mathcal{X}}(x) = \lg(2 \cdot i_x - 1)$ and randomly assigning an ordering to the uniform distribution does not produce an expected value of $\lg N$, but $\approx \lg N - (\lg e - 1)$.

Proof. We first take the expectation:

$$\begin{aligned}
E\left[S^{\mathrm{I}}_{\mathcal{X}}(x) \middle| x \overset{\mathrm{R}}{\leftarrow} \mathcal{U}_N\right] &= \sum_{i=1}^{N} \frac{1}{N} \cdot \lg(2 \cdot i - 1) \\
&= \frac{1}{N} \cdot (\lg 1 + \lg 3 + \lg 5 + \ldots + \lg(2N-1)) \\
&= \frac{1}{N} \cdot \lg(1 \cdot 3 \cdot 5 \cdot \ldots \cdot (2N-1)) \\
&= \frac{1}{N} \cdot \lg\left(\frac{(2N)!}{2 \cdot 4 \cdot 6 \cdot \ldots \cdot 2N}\right) \\
&= \frac{1}{N} \cdot \lg\left(\frac{(2N)!}{2^N \cdot N!}\right)
\end{aligned}$$

We can use Stirling's approximation $\ln N! \sim N \ln N - N$ [130], converting the base to get $\lg N! \sim N \lg N - N \lg e$:

$$\begin{aligned}
E\left[S^{\mathrm{I}}_{\mathcal{X}}(x) \middle| x \overset{\mathrm{R}}{\leftarrow} \mathcal{U}_N\right] &= \frac{1}{N} \cdot \lg\left(\frac{(2N)!}{2^N \cdot N!}\right) \\
&= \frac{1}{N} \cdot \left(\lg(2N)! - \lg N! - \lg 2^N\right) \\
&\approx \frac{1}{N} \cdot (2N \lg 2N - 2N \lg e - N \lg N + N \lg e - N) \\
&= \frac{1}{N} \cdot (2N \lg N + 2N - 2N \lg e - N \lg N + N \lg e - N) \\
&= \frac{1}{N} \cdot (N \lg N + N - N \lg e) \\
&= \lg N - (\lg e - 1)
\end{aligned}$$

□

Appendix C

PIN survey detail

The following is a summary of the presentation and responses in our PIN survey.

Do you regularly use a PIN number with your payment cards?($N = 1337$)

yes, a 4-digit PIN	yes, a PIN of 5+ digits	no
1108 (82.9%)	69 (5.2%)	160 (12.0%)

When making purchases in a shop, how do you typically pay?($N = 1177$)

I use my payment card and key in my PIN	477 (40.5%)
I use my payment card and sign a receipt	357 (30.3%)
I use my payment card with my PIN or my signature equally often	184 (15.6%)
I normally use cash or cheque payments and rarely use payment cards	159 (13.5%)

Overall, how often do you type your PIN when making a purchase in a shop? And how often do you type your PIN at an ATM/cash machine?($N = 1177$)

	shop		ATM	
Multiple times per day	81	(6.9%)	14	(1.2%)
About once per day	117	(9.9%)	19	(1.6%)
Several times a week	342	(29.1%)	118	(10.0%)
About once per week	241	(20.5%)	384	(32.6%)
About once per month	113	(9.6%)	418	(35.5%)
Rarely or never	283	(24.0%)	224	(19.0%)

How often do you use your PIN to unlock a unicorn shed? Since this doesn't make sense, please select 'several times a week' to show that you have been reading carefully.($N = 1351$)

several times a week	other (further responses discarded)
1337 (99.0%)	14 (1.0%)

How many payment cards with a PIN do you use?($N = 1177$)

1	2	3	4	5	6
708 (60.2%)	344 (29.2%)	89 (7.6%)	23 (2.0%)	11 (0.9%)	2 (0.2%)

Median: 1, Mean: 1.5

If you have more than one payment card which requires a PIN, do you use the same PIN for several cards?($N = 469$)

yes	no
161 (34.3%)	308 (65.7%)

Have you ever changed the PIN associated with a payment card?($N = 1177$)

Never	Yes, when I initially received the card	Yes, I change periodically
591 (50.2%)	376 (31.9%)	210 (17.8%)

Have you ever forgotten your PIN and had to have your financial institution remind you or reset your card?($N = 1177$)

yes	no
186 (15.8%)	991 (84.2%)

Have you ever shared your PIN with another person so that they could borrow your payment card?($N = 1177$)

spouse or significant other	475	(40.4%)
child, parent, sibling, or other family member	204	(17.3%)
friend or acquaintance	40	(3.4%)
secretary or personal assistant	1	(0.1%)
any	**621**	**(52.8%)**

Have you ever used a PIN from a payment card for something other than making a payment or retrieving money?($N = 1177$)

password for an Internet account	180	(15.3%)
password for my computer	94	(8.0%)
code for my voice-mail	242	(20.6%)
to unlock the screen for mobile phone	104	(8.8%)
to unlock my SIM card	29	(2.5%)
entry code for a building	74	(6.3%)
any	**399**	**(33.9%)**

Do you carry any of the following in your wallet or purse?($N = 415$)[1]

driver's license	377	(90.8%)
passport or government ID card	68	(16.4%)
social security or other insurance card	155	(37.3%)
school or employer ID listing date of birth	23	(5.5%)
other document listing date of birth	78	(18.7%)
any item with date of birth	**411**	**(99.0%)**

[1]This question was sent to a random subset of respondents after the main survey.

Did you choose your main PIN number yourself? (N=603)

Yes, I chose the number which I use as my main PIN.	433	(71.8%)
No, my main PIN was assigned by my bank.	160	(26.5%)
No, my main PIN was assigned to me by a previous bank.	10	(1.7%)

When choosing your PIN, what did you have in mind? (N=433)

My PIN represents a date and/or year.	115	(26.6%)
My PIN represents a pattern on the keypad.	44	(10.2%)
I chose my PIN randomly or used a number I already memorised.	274	(63.9%)

Does your PIN represent... (N=115)

a 4-digit year (for example, `1952` or `2006`)	24	(20.9%)
a day of the year (for example, `0305` for March 5th	36	(31.3%)
a complete calendar date (for example, `3596` for March 5th, 1996)	38	(33.0%)
some other type of date	17	(14.8%)

Does your PIN... (N=44)

include a straight horizontal line on the keypad	3	(7.0%)
include a straight vertical line on the keypad	3	(7.0%)
include a diagonal line on the keypad	5	(11.6%)
use the 4 corners of the keypad	1	(2.3%)
use the 4 points of the cross on the keypad	2	(4.7%)
use 4 numbers which make a 2 by 2 box on the keypad	0	(0.0%)
represent some other contiguous path on the keypad	7	(16.3%)
spell a memorable word using the letters on the keypad	9	(20.9%)
represent another pattern on the keypad	13	(30.2%)

Did you.. (N=271)

repeat a two-digit number twice (for example `4545`)?	12	(4.4%)
repeat the same digit four times (for example `8888`)?	1	(0.4%)
count a sequence of consecutive numbers (for example `1234`)?	1	(0.4%)
begin or end with 69 (for example `6920` or `4369`)?	4	(1.5%)
choose your PIN randomly yourself, for example by rolling dice?	38	(14.0%)
keep a random PIN assigned to from a previous payment card?	5	(1.8%)
choose a PIN based on your lucky numbers or lottery numbers?	14	(5.2%)
choose your PIN based on another number assigned to you at some point, such as a phone number or ID number?	60	(22.1%)
use some other method to choose your PIN?	136	(50.2%)

Respondents provided free-form feedback on their PIN selection strategies which was often more detailed than their survey responses. For example, many users indicated they used "some other method" then wrote they used a method we had specified. We manually classified approximately 150 users based on their text feedback, which is why the numbers presented in §4 are different for some strategies than the numbers listed here.

Appendix D

List of password data sets

The following data sets are referenced by name in the text. All except the YAHOO set were obtained after a database compromise led to the public disclosure of password records. For each data set, M is the total number of observed passwords in the sample, $V(M)$ is the number of distinct passwords observed and $V(1, M)/M$ is the proportion of observations which were unique in the sample (§5).

70YX		www.70yx.com		
year	hash	M	$V(M)$	$V(1, M)/M$
2011	**none**	**9072966**	**3462284**	**31.3%**

70yx is a gaming website based in China. This data set is one of several large leaks from Chinese websites at the end of 2011. No password restrictions appear to have been in place, but the vast majority of passwords were entered using only ASCII characters and not Chinese characters in UTF-8. The source of leak is unknown, but it is assumed to be the result of a SQL injection attack.

BHEROES		www.battlefieldheroes.com		
year	hash	M	$V(M)$	$V(1, M)/M$
2011	**MD5**	**548774**	**423711**	**69.4%**

Battlefield Heroes is an online multiplayer shooting game. This data set represents users who signed up to participate as beta testers. It appears that a six-character minimum length requirement was in place for all passwords. It was released as part of the "50 Days of Lulz" bundle in 2011.

Gawker		www.gawker.com		
year	hash	M	$V(M)$	$V(1,M)/M$
2010	`crypt()`	**748559**	**620790**	**78.5%**

Gawker is an online blog network which maintains accounts for users to comment on posted stories. This data set was hacked by a group calling itself Gnosis in 2010. There were no restrictions on passwords, though the use of `crypt()` means only the first 8 characters of each password were necessary for login.

Fox				
year	hash	M	$V(M)$	$V(1,M)/M$
2011	**none**	**364**	**348**	**92.9%**

This small data set represents corporate employee accounts at Twentieth Century Fox, a media company. It was released by LulzSec in 2011 as part of the "50 days of Lulz" release. No password restrictions appear to have been in place.

Hebrew		www.wonder-tree.com		
year	hash	M	$V(M)$	$V(1,M)/M$
2011	**none**	**1252**	**1110**	**84.5%**

Wondertree is an Israeli, Hebrew-language religious website. This data set was accidentally posted online by site administrators. There do not appear to have been any password restrictions in place. Though most passwords were entered in the ASCII subset of UTF-8, a small number ($\approx 2.5\%$) were entered using characters from the Hebrew alphabet.

hotmail		www.hotmail.com		
year	hash	M	$V(M)$	$V(1,M)/M$
2009	**none**	**9536**	**8504**	**83.5%**

Hotmail is a webmail platform owned and operated by Microsoft. This data set appears to be part of of a larger set, as all email addresses start with the letters 'a' or 'b'. According to Microsoft the passwords were obtained by phishing. A 6-character minimum length appears to have been enforced.

myBart		www.mybart.org		
year	hash	M	$V(M)$	$V(1,M)/M$
2011	**none**	**2002**	**2002**	**100.0%**

MyBart is the customer relations website for the BART mass transit system in the San Francisco Bay Area. It was hacked by Anonymous in 2011. Of note, many passwords were randomly generated by MyBart in the original system design and about two-thirds of users appear to have kept their random passwords. For the users who changed, no restrictions appear to have been in place.

MySpace		www.myspace.com		
year	hash	M	$V(M)$	$V(1,M)/M$
2006	**none**	**49711**	**41543**	**75.4%**

MySpace is an online social network. These accounts were apparently compromised by a phishing attack. A password policy appears to have been in place prohibiting passwords which only contain numbers or only contain letters. This data set was an early leak and has been studied in several published works [267, 81].

NATO-Books		www.nato.int/cps/en/natolive/e-bookshop.htm		
year	hash	M	$V(M)$	$V(1,M)/M$
2011	**none**	**11524**	**10411**	**85.5%**

This data set represents users of an online bookshop run by NATO, the North Atlantic Treaty Organization. Accounts were used to access publications created by NATO. No password restrictions appear to have been in place. It was released as part of the "50 Days of Lulz" bundle in 2011.

RockYou		www.rockyou.com		
year	hash	M	$V(M)$	$V(1,M)/M$
2009	**none**	**32575653**	**14316979**	**36.4%**

RockYou is a large social media company specialising in application development for social networks. Its servers were hacked via SQL injection and a list of passwords was leaked in plaintext. The company is based in the USA and only offers its content in English At the time of the leak, RockYou claimed to have 65 M registered users, meaning over half were not included in the leak. No password restrictions appear to have been in place. All UTF-8 characters were allowed and the data set does include many passwords in foreign languages.

Sony-BMG				
year	hash	M	$V(M)$	$V(1,M)/M$
2011	**none**	**40459**	**30245**	**63.0%**

This data set represents users of several internal systems for Sony BMG, a large media corporation. No password restrictions appear to have been in place. It was released as part of the "50 Days of Lulz" bundle in 2011.

Yahoo		www.yahoo.com		
year	hash	M	$V(M)$	$V(1,M)/M$
2011	**see §6.1**	**69301337**	**33895873**	**42.5%**

Yahoo! is a large and diversified international web services company offering email, chat, news, social networking, gaming and many other features. This data set was collected from Yahoo! with explicit cooperation (§6.1.5). Each password in this set was hashed with a strong random value which was discarded after collection, preventing analysis of the plaintext passwords. Some of the users registered passwords with a six-character minimum length enforced (§6.2).

Appendix E

Sources of census data

In §7, we made use of data from many government census agencies around the world:

Chile Civil Identification and Registration Service (http://www.registrocivil.cl/)

Des Moines Register (http://data.desmoinesregister.com/petnames/)

Eeski Ekspress (http://paber.ekspress.ee/viewdoc/104BE2FD5331F393C225743300602476)

Euromonitor International (http://www.euromonitor.com/Top_150_City_Destinations_London_Leads_the_Way)

Finland Population Register Center (http://www.vaestorekisterikeskus.fi/vrk/home.nsf/pages/index_eng)

Intellectual Property Australia (http://pericles.ipaustralia.gov.au/atmoss/falcon_search_tools.Main?pSearch=Surname)

Japanese Surname Dictionary (http://park14.wakwak.com/~myj/)

Los Angeles Department of Animal Licensing (http://projects.latimes.com/dogs)

San Francisco Animal Licensing Department (http://www.sfgate.com/webdb/petnames/)

Scottish Government School Education Statistics (http://www.scotland.gov.uk/Topics/Statistics/Browse/School-Education/pupnum2008/)

Statistics Belgium (http://statbel.fgov.be/)

Statistics Faroe Islands (http://www.hagstova.fo)

Statistics Iceland (http://www.statice.is/Pages/847)

Statistics Korea (http://kostat.go.kr)

Statistics Norway (http://www.ssb.no/navn_en/)

United Kingdom Department for Children, Schools, and Families (http://www.edubase.gov.uk/home.xhtml)

United Kingdom Office for National Statistics (http://www.statistics.gov.uk)

United States Census Bureau (http://www.census.gov)

United States Social Security Administration(http://www.ssa.gov/)