Defining Password Strength

Jeffrey Goldberg

Chief Defender Against the Dark Arts
AgileBits Inc.
jeff@agilebits.com

Passwords13

Outline

Fools rush in ...

...where angels fear to tread

Fools rush in where angels fear to tread.

Reasons *To* Define Password Strength

- We informally talk about password strength all the time. E.g.
  - "`olByWo9yIFp8NfOxSprJXX` *is a stronger password than* `Password1`."
  - "*12345? That's terrible, that is the kind of thing an idiot would put on his luggage.*"
- We like to compare password strength to other parts of the system. E.g.,
  - "*Should you really be worrying of the difference between 128-bit AES keys and 256-bit AES keys when your password is probably less than 40 bits?*"

Reasons *To* Define Password Strength

- We informally talk about password strength all the time. E.g.
  - "*`olByWo9yIFp8NfOxSprJXX` is a stronger password than `Password1`.*"
  - "*12345? That's terrible, that is the kind of thing an idiot would put on his luggage.*"
- We like to compare password strength to other parts of the system. E.g.,
  - "*Should you really be worrying of the difference between 128-bit AES keys and 256-bit AES keys when your password is probably less than 40 bits?*"

Reasons *To* Define Password Strength

- We informally talk about password strength all the time. E.g.
  - "*`olByWo9yIFp8NfOxSprJXX` is a stronger password than `Password1`.*"
  - "*12345? That's terrible, that is the kind of thing an idiot would put on his luggage.*"
- We like to compare password strength to other parts of the system. E.g.,
  - "*Should you really be worrying of the difference between 128-bit AES keys and 256-bit AES keys when your password is probably less than 40 bits?*"

Reasons *To* Define Password Strength

- We informally talk about password strength all the time. E.g.
  - "*`olByWo9yIFp8Nf0xSprJXX` is a stronger password than `Password1`.*"
  - "*12345? That's terrible, that is the kind of thing an idiot would put on his luggage.*"
- We like to compare password strength to other parts of the system. E.g.,
  - "*Should you really be worrying of the difference between 128-bit AES keys and 256-bit AES keys when your password is probably less than 40 bits?*"

Reasons *To* Define Password Strength

- We informally talk about password strength all the time. E.g.
  - "*`olByWo9yIFp8NfOxSprJXX` is a stronger password than `Password1`.*"
  - "*12345? That's terrible, that is the kind of thing an idiot would put on his luggage.*"
- We like to compare password strength to other parts of the system. E.g.,
  - "*Should you really be worrying of the difference between 128-bit AES keys and 256-bit AES keys when your password is probably less than 40 bits?*"

Uniform distribution

A "uniform distribution" is one in which all elements are equally likely.

Uniform

- The result of rolling a single (fair) die
- AES keys generated by non-terrible random number generator
- Passwords generated by a good password generator

Not

- The sum of rolling a pair of dice
- Word frequencies in human language
- Passwords generated by humans

Uniform distribution

A "uniform distribution" is one in which all elements are equally likely.

Uniform

- The result of rolling a single (fair) die
- AES keys generated by non-terrible random number generator
- Passwords generated by a good password generator

Not

- The sum of rolling a pair of dice
- Word frequencies in human language
- Passwords generated by humans

Uniform distribution

A "uniform distribution" is one in which all elements are equally likely.

Uniform
- The result of rolling a single (fair) die
- AES keys generated by non-terrible random number generator
- Passwords generated by a good password generator

Not
- The sum of rolling a pair of dice
- Word frequencies in human language
- Passwords generated by humans

Uniform distribution

A "uniform distribution" is one in which all elements are equally likely.

Uniform

- The result of rolling a single (fair) die
- AES keys generated by non-terrible random number generator
- Passwords generated by a good password generator

Not

- The sum of rolling a pair of dice
- Word frequencies in human language
- Passwords generated by humans

Uniform distribution

A "uniform distribution" is one in which all elements are equally likely.

Uniform
- The result of rolling a single (fair) die
- AES keys generated by non-terrible random number generator
- Passwords generated by a good password generator

Not
- The sum of rolling a pair of dice
- Word frequencies in human language
- Passwords generated by humans

Uniform distribution

A "uniform distribution" is one in which all elements are equally likely.

Uniform
- The result of rolling a single (fair) die
- AES keys generated by non-terrible random number generator
- Passwords generated by a good password generator

Not
- The sum of rolling a pair of dice
- Word frequencies in human language
- Passwords generated by humans

Uniform distribution

A "uniform distribution" is one in which all elements are equally likely.

Uniform
- The result of rolling a single (fair) die
- AES keys generated by non-terrible random number generator
- Passwords generated by a good password generator

Not
- The sum of rolling a pair of dice
- Word frequencies in human language
- Passwords generated by humans

Shannon Entropy, $H(\cdot)$.

Definition (Shannon Entropy)

The entropy, $H(X)$, of a discrete random distribution, X, is

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

where x_i is the ith element of X, and $p(x_i)$ is the probability of selecting x_i.

When Shannon Entropy Is Good ...

...it is very very good.

- It's familiar (almost everyone talks in these terms)
- Its units are bits. Yeah bits!
- It is well understood.
- It is commensurate with other systems.
- Each bit doubles cracking time

Only "good" when passwords are distributed uniformly

When Shannon Entropy Is Good ...

...it is very very good.

- It's familiar (almost everyone talks in these terms)
- Its units are bits. Yeah bits!
- It is well understood.
- It is commensurate with other systems.
- Each bit doubles cracking time

Only "good" when passwords are distributed uniformly

When Shannon Entropy Is Good ...

...it is very very good.

- It's familiar (almost everyone talks in these terms)
- Its units are bits. Yeah bits!
- It is well understood.
- It is commensurate with other systems.
- Each bit doubles cracking time

Only "good" when passwords are distributed uniformly

When Shannon Entropy Is Good ...

...it is very very good.

- It's familiar (almost everyone talks in these terms)
- Its units are bits. Yeah bits!
- It is well understood.
- It is commensurate with other systems.
- Each bit doubles cracking time

Only "good" when passwords are distributed uniformly

When Shannon Entropy Is Good ...

...it is very very good.

- It's familiar (almost everyone talks in these terms)
- Its units are bits. Yeah bits!
- It is well understood.
- It is commensurate with other systems.
- Each bit doubles cracking time

Only "good" when passwords are distributed uniformly

When Shannon Entropy Is Good ...

...it is very very good.

- It's familiar (almost everyone talks in these terms)
- Its units are bits. Yeah bits!
- It is well understood.
- It is commensurate with other systems.
- Each bit doubles cracking time

Only "good" when passwords are distributed uniformly

When Shannon Entropy Is Good ...

...it is very very good.

- It's familiar (almost everyone talks in these terms)
- Its units are bits. Yeah bits!
- It is well understood.
- It is commensurate with other systems.
- Each bit doubles cracking time

Only "good" when passwords are distributed uniformly

But when it is bad ...

...it is horrid

- When the distribution is not a uniform distribution, Shannon entropy can yield meaningless results

  Even with an accurate Shannon entropy value, it would not tell the defender anything about how vulnerable a system would be to an online password cracking attack. [Weir, Aggarwal, Collins, et al. 2010, p. 162]

Guessing Entropy

Guessing Entropy is the average number guesses to find x_i in X when X is sorted by likeliness.

Definition (Guessing Entropy)

The Guessing Entropy, $G(X)$ of a distribution X where the values of X are sorted by decreasing probability, so that if $i > j$ then $p(x_i) \geq p(x_j)$,

$$G(X) = \sum_{i=0}^{\max(R(X))} p(x_i)(i+1)$$

[Following formalization of Cederlöf 2005]

When Guessing: The Good and the Bad

Good Guessing Entropy is rooted in the number of guesses it takes to find a password.

Bad
- Not measured in bits
  (Easy to fix)
- Has the same problems as $H(\cdot)$ with fat headed distributions.
  (Really!)

When Guessing: The Good and the Bad

Good Guessing Entropy is rooted in the number of guesses it takes to find a password.

Bad
- Not measured in bits
  (Easy to fix)
- Has the same problems as $H(\cdot)$ with fat headed distributions.
  (Really!)

When Guessing: The Good and the Bad

Good Guessing Entropy is rooted in the number of guesses it takes to find a password.

Bad
- Not measured in bits
  (Easy to fix)
- Has the same problems as $H(\cdot)$ with fat headed distributions.
  (Really!)

When Guessing: The Good and the Bad

Good Guessing Entropy is rooted in the number of guesses it takes to find a password.

Bad
- Not measured in bits
  (Easy to fix)
- Has the same problems as $H(\cdot)$ with fat headed distributions.
  (Really!)

When Guessing: The Good and the Bad

Good Guessing Entropy is rooted in the number of guesses it takes to find a password.

Bad
- Not measured in bits
  (Easy to fix)
- Has the same problems as $H(\cdot)$ with fat headed distributions.
  (Really!)

Min-entropy

Min-entropy is is based solely on the probability of the most likely value in X.

Definition (min-entropy)

The min-entropy, H_∞, of a distribution, X, is the negative base 2 logarithm of the probability of a most probable value in X.

$$H_\infty(X) = -\log_2 p(x_m)$$

where x_m has the maximum probability in X, that is, $\forall x_i \in X, p(x_m) \geq p(x_i)$

[Min-entropy is a special case of Rényi Entropy, so I use that notation]

Our Best Worst Case

Min-entropy can be useful

Although min-entropy throws away an enormous amount of information about the distribution, it may be the most useful entropy notion when talking about distributions of passwords, as it is based solely on the worst case.

What has a fat head and a long tail?

Both Guessing Entropy and Shannon Entropy fail for talking about password strength when passwords are distributed with a fat head (a few elements that are very likely) and a long thin tail (lots of low probability elements).

An extreme example

Imagine if a value, q_0, shows up 9 times out of 10. The remaining 1 time out of 10, it is one of 2^{512} possibilities. This distribution, Q, has one very likely element, and lots of of unlikely elements.

Definition (Troublesome distribution, Q)

Let Q be a distribution of $2^{512}+1$ values. q_0 has a probability of 0.9 and all of the other values, $q_1 \ldots q_n$, have a probability of $(1-0.9)2^{-512}$.

Q and Entropy Results

What different entropy notions do with Q

Shannon Entropy	$H(Q) \approx 51.66$ bits
Guessing Entropy	$G(Q) \approx 2^{507.6}$ guesses
Min-entropy	$H_\infty(Q) \approx 0.15$ bits

Shannon Entropy and Q

$$
\begin{aligned}
H(Q) &= -\left[0.9\log_2 0.9 + \sum_{1}^{2^{512}} (1-0.9)2^{-512}\log_2\left((1-0.9)2^{-512}\right)\right] \\
&\approx 0.13 + 51.53 \\
&\approx 51.66
\end{aligned}
$$

Guessing Entropy and Q

$$
\begin{aligned}
G(Q) &= p(q_0) + \sum_{i=1}^{2^{512}} p(q_i)(i+1) \\
&= 0.9 + \frac{1-0.9}{2^{512}} \cdot \sum_{j=2}^{2^{512}+1} j \\
&= 0.9 + \frac{0.1}{2^{512}} \cdot \frac{2^{512}(2^{512}+3)}{2} \\
&= 0.9 + \frac{2^{512}+3}{20} \\
&\approx 2^{507.6}
\end{aligned}
$$

The trouble with ignoring the password

Using a single statistic (some form of Entropy) for a distribution

- Throws out too much information
- Aims for the "average" password (under various notions of "average")
- Gets distorted results when the distribution is far from uniform

Instead, we should give up on a statistic for a distribution and look at the strength of a particular password with respect to a distribution.

Guess for a particular password

Definition (Guesses Function, Γ)

$\Gamma(p, X, k)$ is the averages number of guesses that the best algorithm needs to find k in X with probability p, where X is a discrete probability distribution, k is a value in X, and p is a probability $0 \leq p \leq 1$.

The arguments of Γ

Γ is a function of three arguments.

- X The distribution the password is drawn from is crucial to how many guesses are needed to find it.
- k The password you are looking for within the distribution matters for the number of guesses
- p Your target probability of finding the password after a number of guesses.

Definition of password strength

Definition (Password Strength)

The strength of a password, w, with respect to a distribution, X is given by

$$S(w, X) = 1 + \log_2 \Gamma(0.5, X, w)$$

This is just the average number of guesses to have a 50% chance of finding the password, w, in some distribution. It is manipulated to have a result in bits.

Not a Big Deal, But ...

- It is a function of *both* the distribution and the password's place within it.
- Its units are convenient
- We know what we mean when we say Password w_i is stronger than password w_j
- It reflects how difficult it is to crack the password.

Not a Big Deal, But ...

- It is a function of *both* the distribution and the password's place within it.
- Its units are convenient
- We know what we mean when we say Password w_i is stronger than password w_j
- It reflects how difficult it is to crack the password.

Not a Big Deal, But ...

- It is a function of *both* the distribution and the password's place within it.
- Its units are convenient
- We know what we mean when we say Password w_i is stronger than password w_j
- It reflects how difficult it is to crack the password.

Not a Big Deal, But ...

- It is a function of *both* the distribution and the password's place within it.
- Its units are convenient
- We know what we mean when we say Password w_i is stronger than password w_j
- It reflects how difficult it is to crack the password.

Using the Information we need

The strength of a password is a function of *both*

- the distribution
- the password's place within the distribution.

Its units are convenient

Its units are convenient

- When X is a uniform distribution $S(w, X) = H(x)$
- It can be compared to encryption key sizes

Its units are convenient

Its units are convenient

- When X is a uniform distribution $S(w, X) = H(x)$
- It can be compared to encryption key sizes

Its units are convenient

Its units are convenient

- When X is a uniform distribution $S(w, X) = H(x)$
- It can be compared to encryption key sizes

We can talk about relative strength

We know what we mean when we say Password w_i is stronger than password w_j

- when w_i and w_j are drawn from the same distribution
- when w_i and w_j are drawn from different distributions.

Reflects crackability

This definition reflects how difficult it is to crack the password.

Well, at least it tries to. There is stuff it ignores.

Reflects crackability

This definition reflects how difficult it is to crack the password. Well, at least it tries to. There is stuff it ignores.

Password strength meters still suck

- Definition is not constructive
  If anything, this definition of strength reinforces the notion that the only reliable way at present to gauge the strength of a password is to try to crack it.
- But sucky strength meters may still be useful.
  There is some experimental evidence that placing (necessarily sucky) password strength meters in some contexts does improve password choice behavior. (Egelman et al. 2013)

It's okay because I wasn't setting out to build a better password strength meter.

Password strength meters still suck

- Definition is not constructive
  If anything, this definition of strength reinforces the notion that the only reliable way at present to gauge the strength of a password is to try to crack it.
- But sucky strength meters may still be useful.
  There is some experimental evidence that placing (necessarily sucky) password strength meters in some contexts does improve password choice behavior. (Egelman et al. 2013)

It's okay because I wasn't setting out to build a better password strength meter.

Q is contrived

The example distribution, Q, used to illustrate the problem with Shannon and Guessing entropy is contrived.

It's okay because it still illustrates what is a deep problem with those entropy notions when used for passwords.

It's okay because actual distributions may approximate a power-law distribution (Malone and Maher 2011), which is also fat headed. (But I haven't done the math on this.)

Q is contrived

The example distribution, Q, used to illustrate the problem with Shannon and Guessing entropy is contrived.

It's okay because it still illustrates what is a deep problem with those entropy notions when used for passwords.

It's okay because actual distributions may approximate a power-law distribution (Malone and Maher 2011), which is also fat headed. (But I haven't done the math on this.)

Q is contrived

The example distribution, Q, used to illustrate the problem with Shannon and Guessing entropy is contrived.

It's okay because it still illustrates what is a deep problem with those entropy notions when used for passwords.

It's okay because actual distributions may approximate a power-law distribution (Malone and Maher 2011), which is also fat headed. (But I haven't done the math on this.)

"Best Algorithm" Isn't Always Best

Definition of Γ assumes "best algorithm" guesses passwords in order of likeliness. But ...

- Fastest crack times may involve proceeding out of order
  - Some candidates may take more time to check than others
  - May be faster to check groups of related candidates together
- Parallelization makes a hash of taking candidates in sequence

So the proposed definition does not reflect actual, practical cracking technology

It's okay because we shouldn't build our definitions around current technology, which changes rapidly.

"Best Algorithm" Isn't Always Best

Definition of Γ assumes "best algorithm" guesses passwords in order of likeliness. But ...

- Fastest crack times may involve proceeding out of order
  - Some candidates may take more time to check than others
  - May be faster to check groups of related candidates together
- Parallelization makes a hash of taking candidates in sequence

So the proposed definition does not reflect actual, practical cracking technology

It's okay because we shouldn't build our definitions around current technology, which changes rapidly.

We still don't know distribution

- Understanding brains is hard
  Without good models of password choice, our X is still a big unknown.
- Modeling choice outcomes
  But we can avoid modeling choice if we can model observed distributions. E.g., Markov models (e.g., Narayanan and Shmatikov 2005) or Probabilistic Context-free Grammers (e.g., Weir, Aggarwal, Medeiros, et al. 2009).

It's okay because it gave me the opportunity the mention some things that I find very interesting.

We still don't know distribution

- Understanding brains is hard
  Without good models of password choice, our X is still a big unknown.
- Modeling choice outcomes
  But we can avoid modeling choice if we can model observed distributions. E.g., Markov models (e.g., Narayanan and Shmatikov 2005) or Probabilistic Context-free Grammers (e.g., Weir, Aggarwal, Medeiros, et al. 2009).

It's okay because it gave me the opportunity the mention some things that I find very interesting.