(Draft) Corinda: An AI Password Cracker and Strength Estimator

Bernardo A. Rodrigues
bernardoaraujor@gmail.com

May 26, 2017

Glossary

A Universe set of all possible Characters (Alphabet). 3

Γ Sample Space of Strings. 6

$\Psi(\tilde{s})$ Parsing of $\tilde{s}$. 5

$\Psi^{-1}(s_i)$ Concatenation of all s_i. 5

Ξ Set of Models. 5

$\alpha(s)$ Predicate. 4

$\hat{\Lambda}_\Gamma$ Composite Search Space defined by the union of all Composite Search Spaces defined by every m_i in $\mathcal{M}_\Gamma$. 7

$\hat{m}(\tilde{s})$ Critical Composite Model of $\tilde{s}$. 6

λ Search Space. 5

$\mathbb{X}(A)$ Profinite set of all possible Strings over A. 3

$\mathcal{C}(\tilde{m})$ Complexity of $\tilde{m}$. 6

$\mathfrak{S}(\theta_i \mathcal{C}_i)$ Model Strength Estimator of $\hat{m}(\tilde{s})$. 7

$\tilde{\lambda}$ Composite Search Space. 6

$\tilde{s}$ Composite String. 4

$\tilde{m}$ Composite Model. 5

c Character. 3

m Model. 4

s String. 3

1 Introduction

The aim of this paper is to describe an attempt to design a Artificially Intelligent Agent that fully captures all the aspects of Password Hash Cracking. Theoretical foundation relies on Set Theory, First-order Model Theory, and Statistical Inference.

Previous results by Rodrigues et al. [?] suggest that human-biased models generate passwords with distinct statistical patterns, raising the following questions:

- Can a Password's Strength be based on Statistical Samples?
- Can the process of Password Cracking be fully optmized by Strength Estimators?

Among other definitions, we formalize the notions of String, Sample Space, Search Domain, Model, Hash Function, and Model Strength. Strings can be viewed as the Character sets commonly referred as "**Passwords**". Sample Spaces can be viewed as string sets commonly referred as "**leaked plaintext password lists**" [?]. Search Domains can be viewed string sets commonly referred as "**dictionaries**". Models and Composite Models can be viewed as "**cracking attack modes**" [?]. Hash Functions can be viewed as what is commonly referred as "**Secure Hash Algorithms**" [?]. Model Strength Estimators can be viewed as equivalent to the popular notion "**Password Strength**".

The Model Strength Estimator proposed here improves upon the concepts already established by Sahin et al [?], allowing a simpler and more effective metric of Strength. In comparison with Sahin's definition, we argue that using a Real number as Strength Estimator feels more intuitive by allowing intermediary values of Strength. Furthermore, we disagree that a Strength Estimator should depend on which Hash Function is being used and the Computational Power of the Attacker. A Strength Estimator should be a function only of the underlying Model and of a Sample Space. That happens because the computational problem commonly referred as "**Password Cracking**" can ultimately be reduced to "**Computational Complexity vs. Moore's Law**", an eternal tug of war. Here, "**Computational Complexity**" refers to the memory and time restrictions of computing $H(\tilde{s})$, while "**Moore's Law**" refers to the constant acceleration in the growth rate of available Computing Power. There's no way of knowing which is going to be on the winning side tomorrow, so the question of "**how long will the cracking process take**" is purely relative and shouldn't be used in a Strength Estimator.

$\cdots$ SALT

Even though this paper does not take an Information Theoretic approach, a few remarks must be made on that matter. Using Character-based Entropy to Measure Password Strength, as suggested by NIST's Electronic Authentication Guideline [?], gives a false sense of security. Even with the Composition Rules defined by NIST, most users still tend to choose Human-biased Models for Password generation, and Passwords end up having really low Entropy. For

example, the String "@ppl3" is not that different from "apple", and that means there is Redundancy inside the system. The main point here is that the set of possible Symbols for the Entropy calculation should not be limited to individual Characters. Instead, Strings must be viewed as Symbols, in order to allow a better Entropy Estimation.

As a conclusion, we argue that the task of generating a Password should not be relegated to a Human-biased Model. Psychological biases have strong influence in the process, increasing predictability and reducing the effective Model Strength. More appropriate methods of Password generation should be used instead. If the Human being insists in participating in the process, Dicewire [?] could be cited as an appropriate strategy. Memorizing many Strong Passwords is not an easy task, thus human beings should not be responsible for that. Password managing techniques should be used instead.

2 Ethical Implications

3 Definitions

3.1 Character

Let c denote a symbol that belongs to a finite set of symbols A. Let us call c a "Character".

3.2 Alphabet

Let A denote the universe set of all possible Characters. Let us call A an "Alphabet".

$$A = \{c_{i=1}, \cdots, c_N\}$$

E.g.:

A is the ASCII Character set

3.3 String

Let s be a well-ordered finite set of Characters. Let us call s a "String".

$$s = \{c_{i=1}, \cdots, c_M | a_i \in A\}$$

Let $\mathbb{X}(A)$ be the set of all possible Strings over A.

$$\mathbb{X}(A) = \wp(A^{\mathrm{n}}), n \to \infty$$

Here, $\wp(S)$ represents the powerset of S, and A^{n} represents the n'th cartesian power of A.

3.4 Composite String

Let $\tilde{s}$ be a well-ordered finite set of Strings. Let us call it a "Composite String".

$$\tilde{s} = \{s_{i=1}, \cdots, s_N\}$$

For the sake of simplicity, let us use "String" and "Composite String" interchangeably, unless stated otherwise.

3.5 Predicate

Let Predicate $\alpha(s)$ be the characteristic function of the relation between a String and a String Set $\subseteq \mathbb{X}(A)$.

E.g.:

$$\lambda \text{ is a dictionary of English words}$$
$$\alpha(s) \implies s \in \lambda$$

Let us use $\alpha(s)$ and α interchangeably.

3.6 Sentence

Let $\phi(s)$ be a Sentence* formed by one or more Predicates α_i, where the variables can be either a Composite String $\tilde{s}_j$, or its constituent Strings $s_{j,k}$.

$$\phi(s) : \forall s(\alpha_1(s), \cdots, \alpha_N(s))$$

Let us use $\phi(s)$ and ϕ interchangeably.

3.7 Model

Let us define K as the set of Strings s_i and Predicates α_j.

$$K = \{\lambda = \{s_{i=1}, \cdots, s_N\}, \phi = \{\alpha_{j=1}, \cdots, \alpha_M\}\} \text{ ???}$$

Let us define Model m as a Structure of signature K, such that every interpretation of $\phi(s)$ is true according to the Tarski's model-theoretic definition of truth [?]. In the context of First-order Model Theory, s_i is referred as m's First-order Language, and λ is referred as m's Domain*.

$$m \vDash \phi(s)$$

E.g.:

$$\lambda_a \text{ is a dictionary of English words}$$
$$\alpha_a \implies s \in \lambda_a$$
$$\lambda_b \text{ is every possible String starting with an Upper-Case Character}$$
$$\alpha_b \implies s \in \lambda_b$$
$$\phi(s) : \forall s(\alpha_a(s), \alpha_b(s))$$
$$m \vDash \phi(s) \Rightarrow \text{“}Example\text{”} \in \lambda_a \cap \lambda_b$$

3.8 Search Domain

Let the String set λ be the Domain of m. Let us call λ "the Search Domain of m".

3.9 Parsing

Let us define the operation of Parsing $\Psi(\tilde{s})$ as splitting $\tilde{s}$ into an ordered partition set of its members.

$$\Psi(\tilde{s}) = s_{i=1}|\cdots|s_N$$

E.g.:

$$\tilde{s} = \text{“}psword1\text{”}$$
$$\Psi_1(\tilde{s}) = \{\text{“}p\text{”}, \text{“}s\text{”}, \text{“}word\text{”}, \text{“1”}\}$$
$$\Psi_2(\tilde{s}) = \{\text{“}ps\text{”}, \text{“}word\text{”}, \text{“1”}\}$$
$$\Psi_3(\tilde{s}) = \{\text{“}psword\text{”}, \text{“1”}\}$$

3.10 Concatenation

Let us define the operation of Concatenation $\Psi^{-1}(s_i)$ as the inverse of Parsing, such that:

$$\Psi^{-1}(s_{i=1}|\cdots|s_N) = \tilde{s}$$

E.g.:

$$\Psi_1^{-1}(\{\text{“}p\text{”}, \text{“}s\text{”}, \text{“}word\text{”}, \text{“1”}\}) = \text{“}psword1\text{”}$$
$$\Psi_2^{-1}(\{\text{“}ps\text{”}, \text{“}word\text{”}, \text{“1”}\}) = \text{“}psword1\text{”}$$
$$\Psi_3^{-1}(\{\text{“}psword\text{”}, \text{“1”}\}) = \text{“}psword1\text{”}$$

Let us also define the String set $\Psi^{-1}(\tilde{\Lambda})$ as the Concatenation of Search Domains such that:

$$\tilde{s} \in \Psi^{-1}(\tilde{\Lambda} = \lambda_{i=1} \times \cdots \times \lambda_N)|\forall i \in \{1, \cdots, N\}, s_i \in \lambda_i$$

3.11 Composite Model

FIX THIS??

Let Ξ be a well-ordered set of models:

$$\Xi = \{m_{i=1}, \cdots, m_N\}$$

Let us define a composite model $\tilde{m} = \tilde{m}(\Xi)$

E.g.:

The domain λ_a of a Model m_a is the String set of names of American citizens
"$john$" $\in \lambda_a$

The domain λ_b of a Model m_b is the set of Strings with numbers in date format
"310790" $\in \lambda_b$

then:

"$john$310790" $\in \tilde{\Lambda}$

3.12 Composite Search Domain

Let $\tilde{\lambda}$ be the String set defined by the cartesian product between string sets λ_i. Let us call $\tilde{\Lambda}$ "the Composite Search Domain of $\tilde{m}(\Xi)$", where each λ_i is defined by each $m_i \in \Xi$.

$$\tilde{\lambda} = \lambda_{i=1} \times \cdots \times \lambda_N$$

3.13 Complexity

Let $|\tilde{\lambda}|$ be the cardinality (number of elements) of the Search Domain of $\tilde{m}$ (either simple or composite). Let $\mathcal{C}(\tilde{m}) = |\tilde{\lambda}|$ denote the "Complexity* of $\tilde{m}$".

Let us use $\mathcal{C}(m_i)$ and $\mathcal{C}_i$ interchangeably.

3.14 Sample Space

Let Γ be a multiset of random Strings defined by statistical samples. Let us call it a "Sample Space". Let γ_i be the multiset consisting of n_i repetitions of each s'_i in Γ.

$$\Gamma = \{\{\gamma_{i=1}\}, \cdots, \{\gamma_N\}\}$$
$$\gamma_i = \{\tilde{s}'_{i,j=1}, \cdots, \tilde{s}'_{i,n_i}\} | \tilde{s}'_{i,j=1} = \cdots = \tilde{s}'_{i,n_i}$$

3.15 Critical Composite Model

Let $M(\tilde{s})$ be the set of all Composite Models that are able to generate the String $\tilde{s}$.

$$M(\tilde{s}) = \{\tilde{m}_{i=1}, \cdots, \tilde{m}_N\} | \forall \tilde{\lambda}_i, \tilde{s} \in \tilde{\lambda}_i$$

Let $\hat{m}(\tilde{s})$ be the Composite Model that minimizes Complexity $\mathcal{C}(\tilde{m}_i)$. Let us call it "$\tilde{s}$'s Critical Composite Model" (CCM).

$$\hat{m}(\tilde{s}) = \tilde{m} \in M(\tilde{s}) | \forall \tilde{m}_i \in M(\tilde{s}), \mathcal{C}(\tilde{m}) = min(\mathcal{C}(\tilde{m}_i))$$

Let $\mathcal{M}_\Gamma$ be the multiset of Critical Composite Models that are able to generate each and every String in an Sample Space Γ.

$$\Gamma = \{\tilde{s}_{j=1}, \cdots, \tilde{s}_L\}$$
$$\mathcal{M}_\Gamma = \{\hat{m}(\tilde{s}_{j=1}), \cdots, \hat{m}(\tilde{s}_L)\}$$

Let us use $\hat{m}(\tilde{s}_j)$ and $\hat{m}_j$ interchangeably.

Also, let $\hat{\Lambda}_\Gamma$ be the Composite Search Domain defined by the union of all Composite Search Domains of every $\hat{m}_i$ in $\mathcal{M}_\Gamma$.

$$\hat{\Lambda}_\Gamma = \bigcup_{j=1}^{L} \tilde{\lambda}_j$$

3.16 Hash Function

Let $H(\tilde{s})$ denote a one-way function that maps each String $\tilde{s}$ into another String h (called "$\tilde{s}$'s Hash"). Note that because $H(\tilde{s})$ is a one-way function, if we don't know what $\tilde{s}$ is, then there is no way of retrieving $\tilde{s}$ from $H(\tilde{s})$.

$$H : \mathbb{X}(A) \to \mathbb{H}$$
$$H(\tilde{s}) = h$$

Here, $\mathbb{X}(A)$ is the Domain*, and $\mathbb{H}$ is the Codomain of $H(\tilde{s})$. A few examples of Hash Functions are SHA1, SHA256, MD5, and bcrypt.

Let $H(\Gamma)$ denote the list of Hashes of each $\tilde{s}$ in a Sample Space Γ.

3.17 Lookup Table

RAINBOW? If we have a Sample Space Γ and a list of its respective Hashes $H(\Gamma)$, then we can form a new set $(\Gamma, H(\Gamma))$ where each element is an ordered pair $(\tilde{s}, h)$. Let us call $(\Gamma, H(\Gamma))$ a "Lookup Table".

$$(\Gamma, H(\Gamma)) = \{(\tilde{s}_i, h_i)\}, \forall \tilde{s}_i \in \Gamma$$

3.18 Cracking Process

3.19 Model Strength Estimator

Let n_i denote the number of occurances of each CCM $\hat{m}_i$ in $\mathcal{M}_\Gamma$. Let θ_i denote the relative frequency of $\hat{m}_i$ in $\mathcal{M}_\Gamma$.

$$\theta_i = \frac{n_i}{|\mathcal{M}_\Gamma|}$$

Let us define the notion of "Model Strength Estimator" $\mathfrak{S}(\theta_i \mathcal{C}_i)$, such that it captures how likely it is for some String $\tilde{s}$ being used as a guess in a Cracking Process, given $\hat{m}_i$'s relative frequency θ_i in $\mathcal{M}_\Gamma$ and $\hat{m}_i$'s Complexity $\mathcal{C}_i$:

$$\mathfrak{S}(\theta_i, \mathcal{C}_i) = \left(1 - \frac{1}{\log_{10}(\mathcal{C}_i + 9)}\right)^{\theta_i}$$

Note that although $\mathfrak{S}(\theta_i, \mathcal{C}_i)$ is related to a Critical Composite Model, we choose to call it Model Strength Estimator for simplicity. Note also that because $\mathcal{C}_i \geqslant 1$, a constant is added to $\mathcal{C}_i$ in order to keep $\mathfrak{S}(\theta_i, \mathcal{C}_i)$'s Range between 0 and 1.

This metric rewards CCMs with high Complexities and low relative frequencies, classifying them as Strong Models. On the other hand, CCMs with low Complexities and high relative frequencies are classified as Weak Models. Note however that a CCM could have a relatively low Complexity and still be classified as having a Medium-High Strength, if its relative frequency is low. Also, CCMs with high Complexities and high frequencies are also classified as having Medium-High Strengths.

GRAPH??

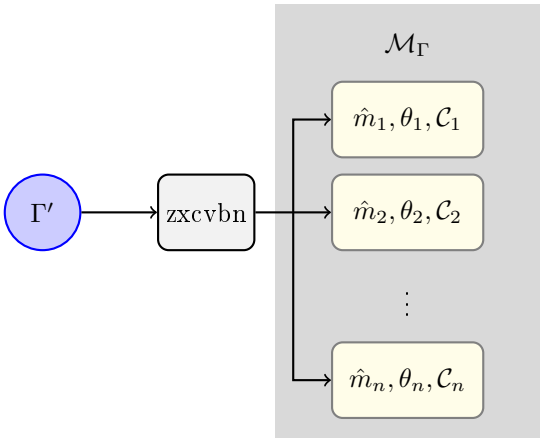
3.20 Cracking Resource

3.21 Corinda's Design

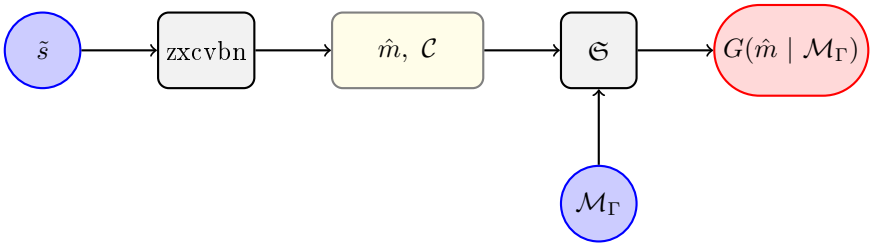
-zxcvbn -hashcat

The Diagram below summarizes Corinda's design:

3.21.1 Training



3.21.2 Passphrase strengh estimator



3.21.3 Hash Cracking