Human-Usable Password Schemas: Beyond Information-Theoretic Security

Elan Rosenfeld — Carnegie Mellon University
Advisor: Manuel Blum — May 4th 2016

Introduction

- People use passwords that are too simple or repetitive[1,2], which are easy for an adversary to break.
- Instead we consider a **password schema**: a mapping from a website name to a password.
- We say a schema has **quality Q** if a *computationally unbounded* adversary can break it with Q **challenge-response pairs**—examples of {website, password}.
- Most prior work focused on theoretical analysis[3,4,5]. This work considers a practical, realistic adversary limited to *currently feasible computation.*

Desiderata for a Good Password Schema

- ***Publishable*** - The schema must be publicly available; the security should only rely on the user's secret key(s).
- ***Human-Usable*** - The schema must be implementable in the user's head, without the use of additional instruments (such as pen and paper).
- ***Secure*** - A *computationally unbounded* adversary who knows the schema should have no better than random chance of being able to correctly guess responses.

Can a computationally bounded adversary be expected to successfully guess the correct response to a new challenge with only Q examples?

To solve this:

1. Generate random challenge-response pairs
2. Build a system of constraints on the user's secret key
3. Use a constraint solver to find a consistent solution

Direct vs. Indirect Schemas

- A challenge-response pair provides sets of possible constraints on the user's secret key.
- For a given challenge length L, the **expansion factor** of a schema is the expected number of possible sets of constraints, denoted F_L.
- We define two categories of password schemas: **direct** and **indirect**. For all direct password schemas, $F_L = 1$. Indirect password schemas are those for which $F_L > 1$.

CONJECTURE:

Any direct, human-usable schema limited to 30 seconds per response can be broken with Q examples by a modern desktop in no more than 24 hours.

- Constraint solvers from as early as 1970 could solve problems with thousands of integer variables[6].
- Complex problems with thousands of integer variables are solvable by today's constraint solvers in less than 20 hours[7].
- With closer to 500 integer variables, these problems can even be solved in as little as a few minutes[8].

Example Schema: Skip-to-my-Lou (STML)

A challenge C consists of L letters $A_1,...,A_L$ and the response consists of m digits $b_1,...,b_m$, $0 \le m \le L$.

Define $f : [A-Z] \to [0-9]$ as a random map from the alphabet to digits.

$STML_f(C)$ denotes the response to C **under STML using secret map** f.

To determine $STML_f(C)$:

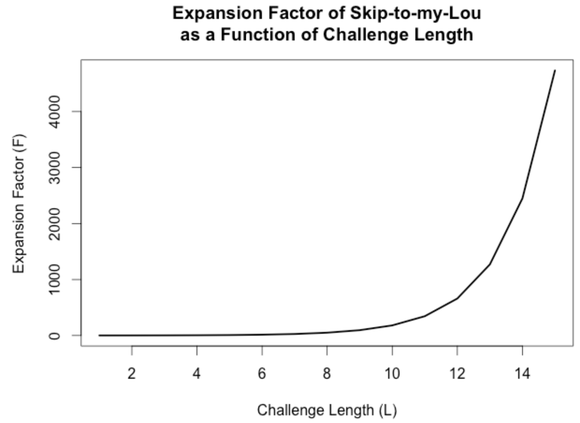
```
Initialize s = 0, j = 0
For i = 1 to L:
    s = (s + f(A_i)) mod 10
    if s ≥ 5:
        Output b_j = s
        j = j + 1
```

- In English: "keep a running total of f applied to the challenge, and only output when the sum (mod 10) is greater than 5."
- An information-theoretic technique for breaking this schema would be to maintain all possibilities for f in a set and eliminate them as inconsistencies arise.
- Using an information-theoretic argument, we can derive an approximate upper bound for Q (namely, $Q \lesssim 8$) for a *computationally unbounded* adversary.
- **Using a CSP solver, we utilize mixed integer linear programming (with an actual computer) to break STML in Q = 7.87 for L = 10.**

- STML is **indirect**—for a given L,

$$F_L = \frac{1}{2^L} \sum_{i=0}^{L} \binom{L}{i}^2$$

- Even with exponential growth of the expansion factor, our DFS algorithm can break STML with L = 10 in less than 5 minutes.
- For L = 15, the algorithm cannot find a solution, despite several days of runtime.
- Need to create a faster algorithm that combines rapid elimination with "Branch and Bound" heuristics to prune tree of constraint combinations closer to the root.



References

[1] L. F. Cranor, "What's wrong with your pa$$w0rd?" (2014, March), [Video file]. Retrieved from https://www.ted.com/talks/lorrie_faith_cranor_what_s_wrong_with_your_pa_w0rd?language=en.
[2] R. Morris and K. Thompson, "Password security: A case history," *Commun. ACM*, vol. 22, no. 11, pp. 594–597, Nov. 1979. [Online]. Available: http://doi.acm.org/10.1145/359168. 359172.
[3] N. J. Hopper and M. Blum, *Advances in Cryptology — ASIACRYPT 2001: 7th International Conference on the Theory and Application of Cryptology and Information Security Gold Coast, Australia, December 9–13, 2001 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, ch. Secure Human Identification Protocols, pp. 52–66. [Online]. Available: http://dx.doi.org/10.1007/3-540-45682-1_4
[4] M. Blum and S. Vempala, "Publishable humanly usable secure password creation schemas," in *Proceedings of the Third AAAI Conference on Human Computation and Crowdsourcing*, 2015. [Online]. Available: https://www.aaai.org/ocs/index.php/HCOMP/HCOMP15/paper/viewFile/11587/11430
[5] J. Blocki, M. Blum, and A. Datta, "Human computable passwords," *CoRR*, vol. abs/1404.0024, 2014. [Online]. Available: http://arxiv.org/abs/1404.0024
[6] M. Benichou, J. M. Gauthier, P. Girodet, G. Hentges, G. Ribiere, and O. Vincent, "Experiments in mixed-integer linear programming," *Mathematical Programming*, vol. 1, no. 1, pp. 76–94, December 1971. [Online]. Available: http://dx.doi.org/10.1007/ BF01584074
[7] J. Zhou, "Computational Experiments for Local Search Algorithms for Binary and Mixed Integer Optimization," Master's thesis, Massachusetts Institute of Technology, September 2010.
[8] V. Jain and I. E. Grossmann, "Algorithms for hybrid milp/cp models for a class of optimization problems," *INFORMS J. on Computing*, vol. 13, no. 4, pp. 258–276, Sep. 2001. [Online]. Available: http://dx.doi.org/10.1287/ijoc.13.4.258.9733