Dynamic Authentication: Developing an Alternative to Passwords (https://dynauth.io)

Connor Peters
Dept. of Computer Sciences
The College at Brockport
Brockport NY, USA
cpete4@brockport.edu

Dr. Ning Yu
Dept. of Computer Sciences
The College at Brockport
Brockport NY, USA
nyu@brockport.edu

Dr. Christine Wania
Dept. of Computer Sciences
The College at Brockport
Brockport NY, USA
cwania@brockport.edu

***Abstract*—As the Internet has matured, the ubiquitous use of passwords as the mechanism to secure user accounts has resulted in passwords becoming a cornerstone of the Internets cybersecurity infrastructure. Unfortunately, overtime a paradox has developed: passwords need to be long and random to be secure, yet easily memorable. This paradox has resulted in a considerable amount of user experience issues that are only getting worse. The purpose of this paper is to propose and implement an alternate dynamic authentication model that addresses some of the problems inherent to passwords. Dynamic authentication (dynauth) is a password replacement scheme that relies on short words, known as keys, to be associated with numbers, known as locks. Before authentication, a service displays a limited number of the users locks in a random order. The user inputs their associated keys as one entry to create a one-time password to authenticate against. Dynauth is intended to be an independent service that utilizes the OAuth 2.0 protocol to authenticate users across domains using a common database. The backend of this design will be developed using Go to present an API to a frontend web-based application as a proof of concept that will then be used to perform usability tests.**

***Index Terms*—cybersecurity, frontend, backend, dynauth**

I. Introduction

Authentication is one of the few aspects of cybersecurity that the average Internet user interacts with directly on a consistent basis. Most people are familiar with the process of creating an online account: you supply your email and a password and then confirm your email to be real.

The purpose of this project is not to expose the insecurities of passwords, as that is already well documented; rather, the purpose of this project is to propose and validate a new knowledge-based authentication scheme as a replacement to the everyday use of passwords.

II. A Brief Review of the Current State of Authentication

There are three basic "factors" of authentication:

- Something you know (knowledge-based)
- Something you have (possession-based)
- Something you are (trait-based)

These are fairly straight-forward and self-explanatory. Something you know refers to a secret that is memorized, ie. a password, passphrase, pin, or security question. Something you have refers to an item you possess such as a key or smartphone. Something you are refers to a biological trait you exhibit, such as your fingerprints or facial structure.

The most commonly used

A. Knowledge-based Authentication

Passwords, password usage, why passwords are so prevalent

B. Possession-based Authentication

Authentication keys (Pico) and stuff

C. Trait-based Authentication

Biometrics, specifically finger prints (touchID) and facial recognition (faceID)

III. The Prevalence and Problems of Passwords

Weren't passwords supposed to die a decade ago [**?**, Need to cite]? Why are they more prevalent now than ever [**?**, Need to cite]? From my observations, there are 3 main reasons:

1) Lack of a viable alternative
2) Familiarity of use
3) Ease of implementation

A. Lack of a Viable Alternative

When accessing any sort of website, people expect passwords. If they were presented anything else[1], they would be confused and not understand what they were seeing. They wouldn't know how to use it, they wouldn't trust it, and they would most likely just leave altogether.

An interesting example of this is my credit card provider's online system. The password requirement are exceedingly stringent, requiring something like 14 characters with all the symbols and goofiness you would expect. On top of that, the bank also requires you to choose an image and 1 of a dozen or so predefined hints[2] to make up for the complex password. Needless to say this has not helped me one whit[3].

[1]With the possible exception of using a smartphone finger print scanner

[2]Some examples are: "similar washer", "unique clothier", "interesting tailor". I have yet to figure out the intended use for such abstract phrases...

[3]Is that a word people still use? Judging by the graph of usage over time Google so generously provides when you research a word definition, no.

Locks	Keys
1	ant
2	beetle
3	cat
4	dog
5	eagle
6	fish
7	goat
8	hare
9	ibis
10	jackal

TABLE I
A SAMPLE TABLE OF LOCKS AND KEYS

B. Familiarity of use

C. Ease of implementation

IV. Enter: Dynamic Authentication

"Dynauth" is a portmanteau of dynamic and authentication, and will be used colloquially to refer to dynamic authentication for the rest of this paper.

A. On The Name

I am willing to admit that "Dynauth" or even "Dynamic Authentication" might not be the most ideal name for such a mechanism due to the ambiguity around the word "dynamic". It was suggested to name it "Active Authentication" due to the fact that the user needs to "actively" think about the process every time they authenticate, reinforcing the memorization of the locks and keys. Despite the nice alliteration, I decided to keep the "dynamic" due to the fact that I had already bought the domain name "dynauth.io" and I did not want to change it.[4]

B. Need for a New Method of Authentication

To be clear, the need for a new *ubiquitous, knowledge-based* method of authentication

C. Basic Traits Necessary to Replace Passwords

Why it needs to be knowledge-based and easy to remember

D. Overview of Usage

A user configures a table of numbered "locks" that correlate to strings of text (typically plain English words) known "keys". These locks and keys are similar to a password in that they are the user's "secret", and must remembered for authentication.

Once the user configure their locks and keys, they are ready to login. A typical login sessions goes like this:

1) The user enters in their username (a valid email address, for this implementation)
2) The user is presented 4[5] of their locks in a random order, without repeat
3) The user inputs, in one long string without spaces or delimiters, the keys that correlate to the randomly chosen locks in the same order

[4]Humans are stubborn

[5]The number 4 was a completely arbitrary number chosen for this implementation simply because it seemed reasonable, both in terms of memorization and security. This number is not set in stone, and more testing will have to be done to determine what the optimal number would be.

E. Example (Correct) Usage 1

Using the same locks and keys as depicted in Table 1, here is what a correct (meaning the keys are inputted correctly by the user) login session might look like:

- Please enter your email:
  - cpete4@brockport.edu
- Your locks are: 7 - 4 - 2 - 10
  - goatdogbeetlejackal
- Correct! You are now authenticated

F. Example (Incorrect) Usage 2

Using the same locks and keys as depicted in Table 1, here is what a incorrect (meaning the keys are inputted incorrectly by the user) login session might look like:

- Please enter your email:
  - cpete4@brockport.edu
- Your locks are: 3 - 6 - 8 - 9
  - catfishharejackal[6]
- INCORRECT: Your keys do not match, not authenticated. Please try again
- Your locks are: 9 - 4 - 2 - 1[7]
  - ibisdogbeetleant
- Correct! You are now authenticated

G. Information Storage

One of the main problems with the usage

V. Benefits of Dynauth

VI. Implementation of Dynauth

How it was implemented

A. Method of Implementation

How I did it and stuff

B. Backend

The backend of this implementation is perhaps the most important aspect of this project because:

1) It provides a testbed to analyze the security benefits of dynauth
2) It provides benchmarks to test out performance in a tangible way
3) It forced me to consider every HTTP request sent over the Internet

The backend of the system was designed as a REST-like[8] API and was written entirely in Golang[9]. The advantages of using Golang

[6]Notice the word "jackal" is for lock number 10, not 9 as required

[7]Notice that the locks changed automatically upon failure.

[8]I describe it as "REST-like" due to the fact that the API is not entirely stateless. Once the user initially sends a login request to retrieve the random locks from the server, those locks are stored in order to be used again during authentication.

[9]https://golang.org/

C. Frontend

An Angular 5 JavaScript application to send HTTP requests to the API

D. JWT Tokens

VII. Challenges During Implementation

What did I encounter while writing all that code?

A. Hashing Speed

Bcrypt was too dang slow, SHA3 and SHA256 had my back

B. Data Storage Problems

Had to create a separate table for each user for speed of access

VIII. Usability Testing

When it comes to authentication, user experience is of paramount importance; It would be trivial to make passwords secure by simply requiring them to be 20+ characters. However, we have learned over time that doing so would not actually result in anyone being more secured due to the compromises that would inflict upon the users [**?**, Need to cite]. Having some sort of authentication scheme that integrates well across domains

A. Method of Usability Testing

Normal password control group and Dynauth 10x4 control group

B. Sample Size

Used the CIS404 class as my testing group

IX. Results of Usability Testing

Results here

A. Instructions Are Important

I forgot to tell people to remember their locks and keys. Great.

X. Conclusion

After a while may suffer the same eventual flaws as passwords

A. Figures and Tables

a) Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation "Fig. 1", even at the beginning of a sentence.

Fig. 1. Example of a figure caption.

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity "Magnetization", or "Magnetization, M", not just "M". If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write "Magnetization (A/m)" or "Magnetization {A[m(1)]}", not just "A/m". Do not label axes with a ratio of quantities and units. For example, write "Temperature (K)", not "Temperature/K".

TABLE II
Table Type Styles

Table Head	**Table Column Head**		
	Table column subhead	***Subhead***	***Subhead***
copy	More table copy[a]		

[a]Sample of a Table footnote.

Acknowledgment

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". Avoid the stilted expression "one of us (R. B. G.) thanks ...". Instead, try "R. B. G. thanks...". Put sponsor acknowledgments in the unnumbered footnote on the first page.

References

Please number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] was the first ..."

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [4]. Papers that have been accepted for publication should be cited as "in press" [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

References

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.
[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
[3] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
[4] K. Elissa, "Title of paper if known," unpublished.
[5] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.
[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.