punchTimeAuth
A Biometric keystroke based authentication

Joy Yang and Don Derek Haddad

Final Report
6.858 Fall 2016

Introduction

Password authentication has been around since the *epoch time*. A system can authenticate its users in so many ways; “The quest to replace passwords” [1] compares these mechanisms looking at three major factors: usability, deployability and security. For instance, a system enhanced with a two-factor authentication (2FA) requires its users to carry a physical proof to login. While this may increase system security, it decreases usability and on deployability. In addition, the security of the system will now also depend on the security of the token’s provider. In order to address these issues, we propose introducing an alternative system that, if necessary, can resort to 2FA. This system analyzes the users’ keystroke time inputs, essentially using muscle memory as an additional physical proof.

We conducted preliminary analyses to better understand the relationship between users and their password inputs and to assess the feasibility of such a system. The plots below show results from 2 trials of 2 users each. In each trial, there is one “user” and one “imposter,” both entering the same password. The axes in these plot are the first two principal components of the typing timing sequences. Here the boundary denotes a quadratic discriminant analysis (QDA) applied on the first two principal components. Even with this reduced information, we are able to obtain fairly good sensitivity as well as specificity in our classification. The intuition for using QDA is that the method utilizes differences in both mean and variance for classification - a user and an imposter will enter the password at different typing speeds, meaning the mean of the data will likely be different; and a user will tend to develop familiarity with the password, making the variance of his/her time sequence data low, while the variance in an imposter’s data will be high.

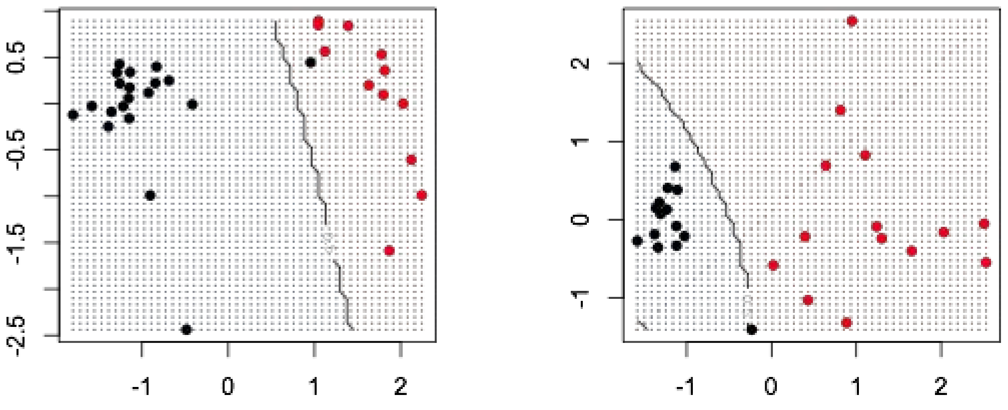


Figure 1: Each point represents a password entry. The black points depict a user’s attempts, and the red points depict an imposter’s attempts. The black line represents the QDA decision boundary.

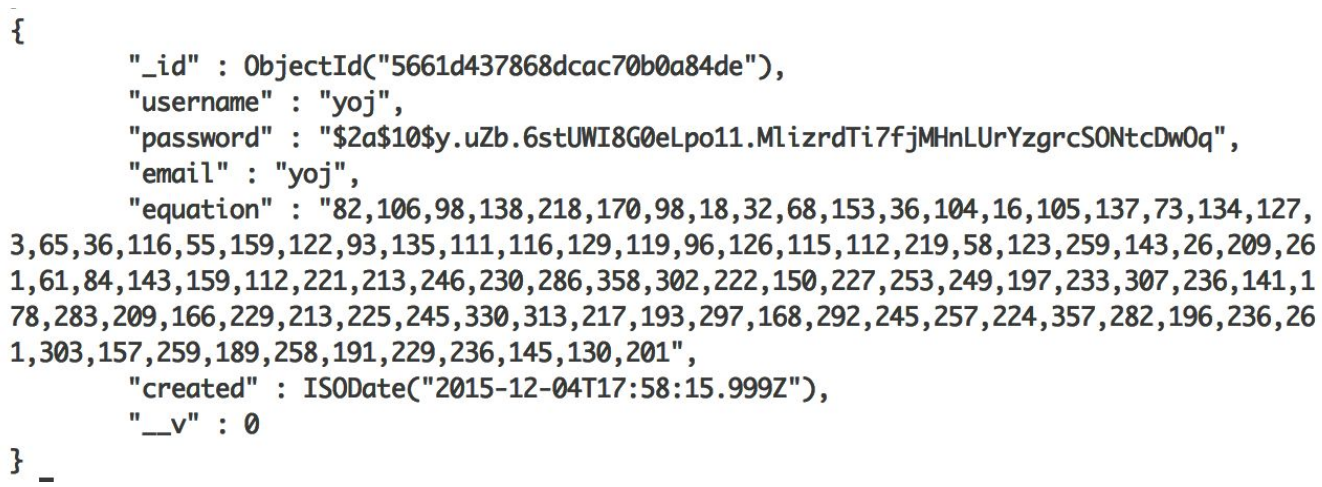
The data was collected from the user’s browser using Javascript onkeypress events and “Date.now()” which returns a timestamp [2]. This preliminary analysis was conducted using R [11].

Related Work

A similar approach was applied to Google's reCaptcha project [3]. Users now will only need to check a checkbox instead of guessing distorted text to prove their non-botness [4]. The reCaptcha project looks at the user's mouse behavior to differentiate a user from a bot. On the other hand, keystroke password biometric authentication has been around since 1980 [5]. Instead of inputting a password, the user is required to write a paragraph to authenticate. punchTimeAuth diverges from this implementation as it looks at identifying a user after inputting a correct password.

Design and Implementation

Upon registration, the user will need to enter a password multiple times to generate a "digital signature." This signature consists of confidence intervals for time differences between keystrokes. Storing this signature as is can reveal the user's password length if the database was compromised by an attacker. With this information, an attacker can brute-force a user's password in a reasonable amount of time. To prevent this disaster from happening, we dilute the user's timestrokes by adding a padding depending on the password's length, here, we use 50 - N as the padding, where N is the length of the password. Fig. 2 shows a record from the MongoDB users collection, password is hashed and salted with bcrypt [8] and the "equation" field stores the time sequences with the padding.

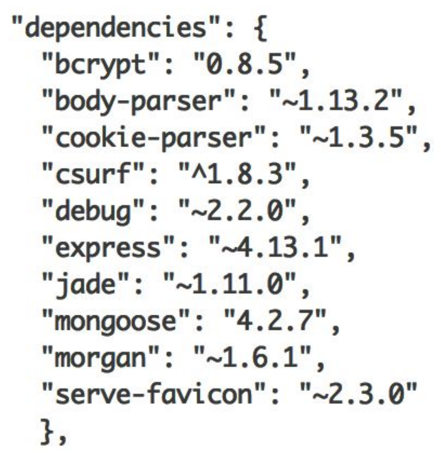


```
{
        "_id" : ObjectId("5661d437868dcac70b0a84de"),
        "username" : "yoj",
        "password" : "$2a$10$y.uZb.6stUWI8G0eLpo11.MlizrdTi7fjMHnLUrYzgrcSONtcDwOq",
        "email" : "yoj",
        "equation" : "82,106,98,138,218,170,98,18,32,68,153,36,104,16,105,137,73,134,127,
3,65,36,116,55,159,122,93,135,111,116,129,119,96,126,115,112,219,58,123,259,143,26,209,26
1,61,84,143,159,112,221,213,246,230,286,358,302,222,150,227,253,249,197,233,307,236,141,1
78,283,209,166,229,213,225,245,330,313,217,193,297,168,292,245,257,224,357,282,196,236,26
1,303,157,259,189,258,191,229,236,145,130,201",
        "created" : ISODate("2015-12-04T17:58:15.999Z"),
        "__v" : 0
} _
```

Figure 2: Sample record from MongoDB

Software Stack:

The project was implemented using Node.js and MongoDB on the backend. Fig.3 shows the list of node modules used extracted from the package.json file. The web server is built with Express.js, a web application framework for Node.js. We also hardened POST requests against cross-site request forgery attacks using the csurf module [7].



```
"dependencies": {
  "bcrypt": "0.8.5",
  "body-parser": "~1.13.2",
  "cookie-parser": "~1.3.5",
  "csurf": "^1.8.3",
  "debug": "~2.2.0",
  "express": "~4.13.1",
  "jade": "~1.11.0",
  "mongoose": "4.2.7",
  "morgan": "~1.6.1",
  "serve-favicon": "~2.3.0"
  },
```

Figure 3: Node modules used in the implementation

The client side was built using Javascript. Because in reality, at registration, users may find it difficult to input “imposter sequences,” the ideal classification scheme would be unsupervised, with no “imposter” datapoints. A few different methods were attempted, including a distance based approach, an initial dimensionality reduction using principal components analysis, and a simple outlier-based approach. In the end, the simple outlier-based approach was used, because yielded the best results during our “product testing.”

The method is as follows: observations from registration are used to create boundaries outside which the particular time difference is called an outlier. Then, if over 10% of these time differences are called outliers, login is refused. These boundaries are (Q1-4*IQR, Q3+4*IQR), where Q1 and Q3 are the first and third quartiles of the particular time difference, and IQR is the average interquartile range over all time differences. The quartiles were calculated using the seventh quantile method described in Hyndman and Fan [11]. This is the default used by R and S.

Figure 4 shows the registration and the authentication forms respectively. A user must enter his password correctly, 5 times in order to train the model. Upon authentication the classical user password matching is applied before matching keystrokes data.

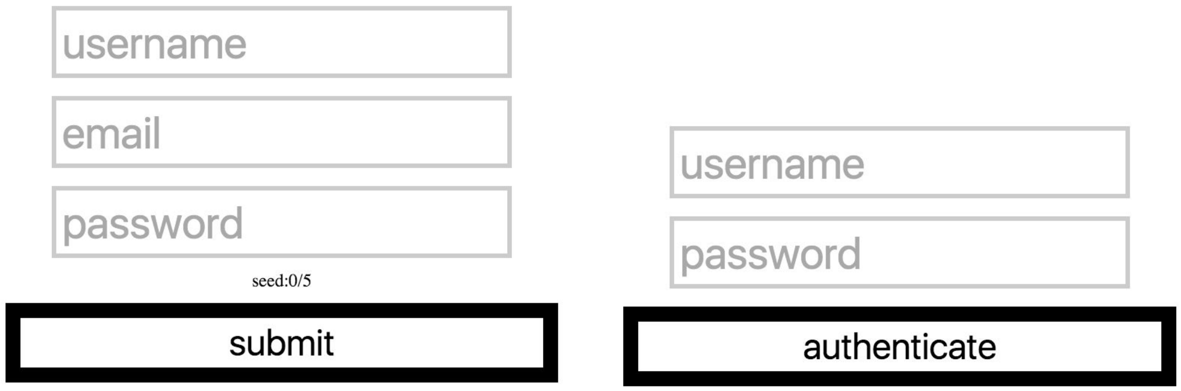


Figure 4: Registration and Authentication forms

Conclusion

In this project we address the practical advantages of using keystroke dynamics as an additional physical proof for authenticating a user. punchTimeAuth investigated the process of analyzing the way users type by measuring keyboard input times and authenticating them based on habitual patterns in typing rhythm.

References

1. Bonneau J, Herley C, Oorschot PC van, Stajano F (2012) The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In: 2012 iEEE symposium on security and privacy. IEEE, pp 553–567

2. Date.now() - JavaScript | MDN.

3. Google Online Security Blog: Are you a robot? Introducing “No CAPTCHA reCAPTCHA”.

4. Google Online Security Blog: Street View and reCAPTCHA technology just got smarter.

5. Choraś RS (2010) Image Processing and Communications Challenges 2. doi: 10.1007/978-3-642-16295-4

6. Araujo L, Sucupira L, Lizarraga M, et al. (2005) User authentication through typing biometrics features. IEEE Transactions on Signal Processing 53:851–855. doi: 10.1109/TSP.2004.839903

7. Joyce R, Gupta G (1990) Identity authentication based on keystroke latencies. Communications of the ACM 33:168–176. doi: 10.1145/75577.75582

8. A Future-Adaptable Password Scheme.

9. dougwilson (2015) Csurf - cSRF token middleware. npm, Vienna, Austria

10. R Core Team (2015) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Oakland, California

11. Hyndman RJ, Fan Y (2012) Sample Quantiles in Statistical Packages. The American Statistician