Neural Network Typing Authentication TJHSST Senior Research Project Research Paper Computer Systems Lab 2009-2010

Luke Knepper
lukeknep@stanford.edu

June 11, 2010

Abstract

This goal of this project was to develop and test a system which authenticates users based on their typing characteristics. This project examines the effectiveness of a neural-network approach for typing characteristic authentication. Log-in-time authentication and continuous authentication based on the user's typing characteristics were both explored, and several demo systems were developed. This process will be beneficial to user security because it will ensure that an intruder does not gain wrongful access to a user's account.

Keywords: authentication, security, typing patterns, neural networks

1 Introduction

The purpose of this project was to develop and test a system which authenticated users using neural network algorithms by use of their typing characteristics. Authentication involves either allowing or denying access to a user when the user attempts to access a system. The goal of authentication techniques are to correctly grant access to a user attempting to access his/her own account or correctly deny access to a user attempting to access

an account that is not his/hers with the highest frequency possible. Since typing characteristics differ by person, they are a biometric which can be used effectively, efficiently, and inexpensively to provide another layer of authentication security for users. Neural networks are algorithms which detect patterns in large amounts of data, and can be used to measure the similarity between different sets of typing data. Different neural network techniques and conditions can affect the accuracy of the neural network at correctly authenticating the user, i.e. either granting the user access when he/she attempts to access his/her own account, or denying the user access when he/she attempts to access an account that is not his/hers.

2 Background

Current authentication techniques span all three tiers of security:

- Tier 1 - Identification (usernames)
- Tier 2 - Knowledge and Possession (passwords, ID cards, security questions, etc.)
- Tier 3 - Skills and Capabilities (captchas, voice recognition)

In the online world, usernames can be stolen, passwords and security questions can be guessed, and captchas can be cracked. However, authentication using a user's typing characteristics, a tier 3 security method, does not have these weaknesses and further has numerous advantages over the other approaches. Authentication through analysis of typing characteristics has been previously proposed, such as U.S. Pat. No. 6,151,593 by Cho, et al (2000). The approach presented herein analyzes the typing characteristics when the user enters that dynamically generated content into the system, thus creating a biometric for that user which is compared to a previously captured biometric for that user. The comparison is performed using a neural network which is trained using the previously-measured typing characteristics. The results of the comparison are compared to a predetermined threshold to determine if the user will be granted access to the system.

The authentication will be primarily done using neural network methods. Neural networks are based off of the way that the human brain works: they are made up of a bunch of nodes (like the brain's neurons) which have inputs, weights, and outputs. Each node collects its inputs, performs a simple

calculation on them (such as a sum), multiplies the total by its weight, and then outputs the final total to the next node(s). The network starts by an input vector (in this case the typing data) which gets sent to the first level (or hidden layer) of nodes and passed down throughout the rest of the layers. The final layer is a single node which outputs a value between 0 and 1, with 1 being a success (or allowed access) and 0 being a failure (or a rejection). A predetermined threshold determines how the final value should be treated (i.e. if the value is above the threshold, it is treated as a 1, and if the value is below the threshold, it is treated as a 0).

There is currently a patent (US 6151593) for an authentication scheme by typing pattern analysis. This method reads in the time between keystrokes for a user when typing their password and then trains a three-layered neural network to this combination. This is the most basic application of typing techniques for authentication. Further, multiple typing pattern log-in software packages exist, such as Psylock. This project will develop a similar authentication algorithm as these products use and then test the effectiveness of this algorithm and analyze the neural network to optimize the network structure.

Another team working under L. Maisuria compared the accuracy of neural networks compared to cluster algorithms. A multi-layered perceptron-based neural network which learned on the Hebbian learning theory was used, as were ten different metrics to compare the clusters for the clustering. They tested the different algorithms by recruiting twenty volunteers to participate in three different sittings. In the first sitting, they all trained their neural networks by typing in their password sixty times. In the next sittings, they attempted to log in to their accounts and break into the accounts of others. The sittings were spaced out by one week. The study found that the clustering methods were slightly more accurate than the neural networks in rejecting impostors. They only found an average of 80% to 90% accuracy in rejection rate, not enough to comprise a stand-alone security system but certainly good enough to be used in conjunction with traditional methods. They found that all keystrokes should be measured, including the beginning and end strokes to the enter key, for the highest accuracy. They found that allowing impostors to observe the users typing before attempting to break in to their accounts had little effect on the accuracy.

An independent team of researchers, headed by Peacock et al., tested the effect of many variations, including neural network set-up, password length, acceptance stringency, data used, and function used. They found the most

effective neural network structure from their tests was to use a set-up where many independent neural networks are trained on different cores (i.e. parallel processing) using randomly generated starting weight vectors. During the training, the best weight vectors are picked and created using genetic algorithms. They found the smaller (more stringent) acceptance ranges came up with a good amount of false alarms (the system incorrectly denied access to the accounts owner 22% of the time) but also minimized break-ins (wan impostor was incorrectly granted access 3% of the time). They also found the most effective password length was 7 characters, a mid-sized password (the longer passwords had no break-ins but many false alarms, and the shorter passwords had many break-ins). They concluded that a linear evaluation function was more effective than a quadratic function and that averaging was more effective than counting each training run. They suggest their results can be improved (75% success, 22% false alarm and 3% break-in for their best algorithm).

2.1 Development and Testing

The following is a simple summary of the various procedures used in this project, and following it are more detailed records:

A log-in simulation program was created to demonstrate the typing authentication procedure. The user can create an account or log-in to an existing account. They specify their username and then type the given sentence and a comfortable pace while the program measures their typing data. If creating an account, the system then trains a neural network with their typing data. If logging in to an account, the system then computes the network value for their data, and if this value is within the given threshold of 1, they are admitted, otherwise they are denied access.

A continuous authentication simulation program (Fig. 2) was developed to show another application of the authentication technique. The user types a sentence while the program measures their typing data and uses the data to train a neural network. The user then interacts with the program in an instant-messaging format. After each message, the program runs the user's typing characteristics through the network, and if the value does not meet the given threshold the warning level is raised. Once the warning level reaches a critical value the user is locked out.

The neural network compute algorithm works as follows:

1. Generate data vector from typing data

2. For each node, top-down, multiply that node's input(s) by that node's weight
3. Sum the nodes of the final layer and put the sum into the activation function
4. Return the final value, a number between 0 and 1

The neural network training algorithm works as follows:
1. Generate L node levels with random weights
2. Randomly select N data files from the data set* and generate data vectors for these files, as well as for the user's typing data
3. For C cycles on every selected data set:
i. Alternate computing the network value for the user's data vector and one of the other randomly selected data vectors.
ii. Change the current weight (given by the cycle number) by a small amount
iii. Compute the network value again, and calculate the error from the desired output value (1 for the user's data, 0 for the non-user's data)
iv. Adjust the current weight accordingly to minimize the error

(*: In the second quarter, a data collection program was developed, was posted online, and collected 1,600 sample data files.)

3 Drawings

FIG. 1 is a mockup of an account set-up screen for a computational system.

FIG. 2 is a flowchart for training the authentication system.

FIG. 3 is a flowchart of the continuous authentication algorithm.

FIG. 4 is the GUI of the mock-login application

FIG. 5 is the GUI of the data collection applet

FIG. 6 is the GUI of the continuous authentication simulation program

FIG. 1: STAGE 1

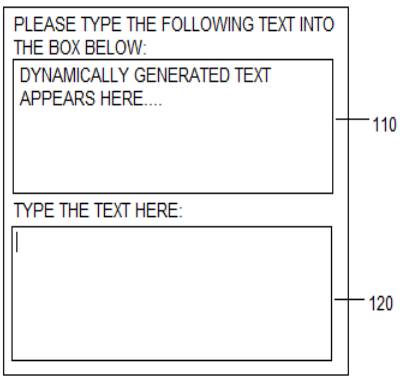


FIG. 2: TRAINING ALGORITHM FLOWCHART

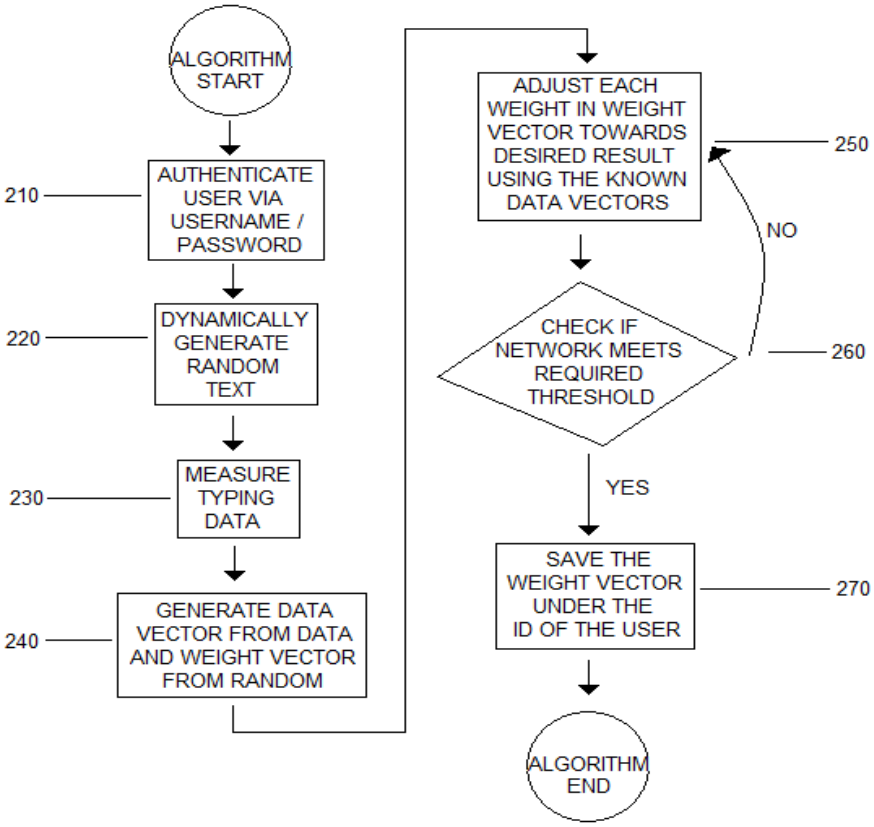


FIG. 3 : AUTHENTICATION ALGORITHM FLOWCHART

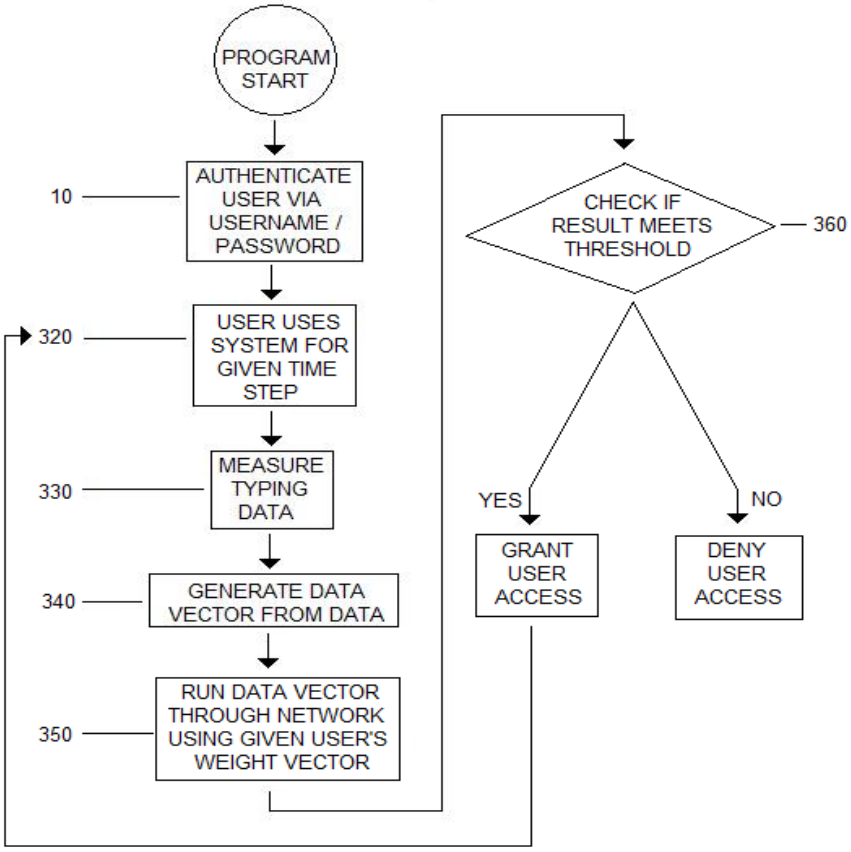


FIG. 4:

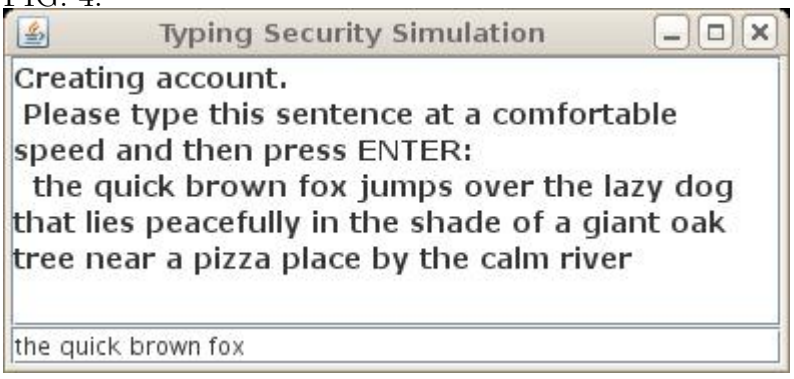


FIG. 5:

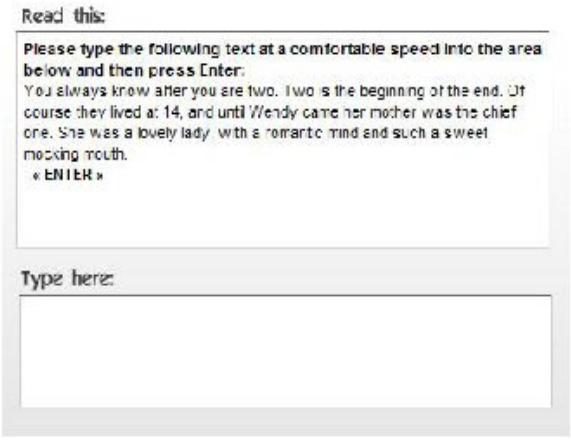
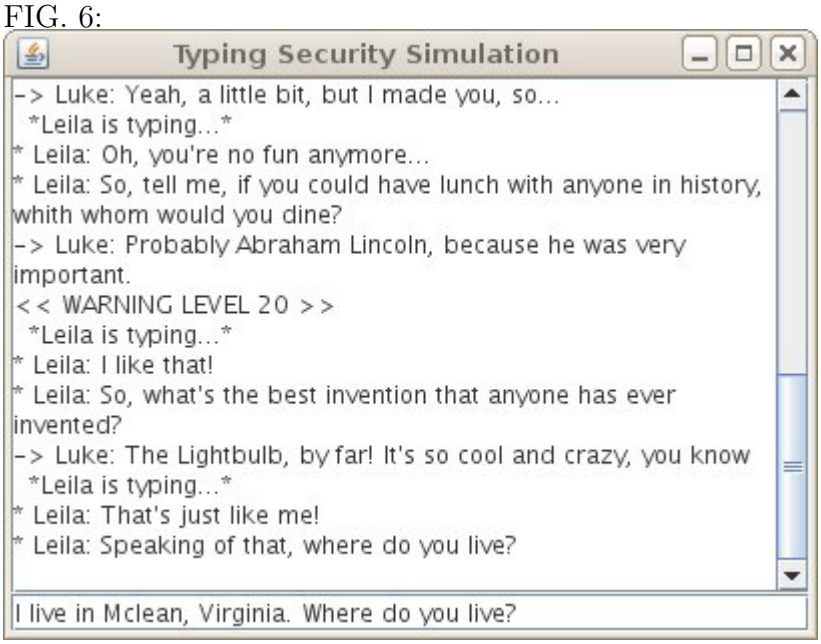


FIG. 6:



4 Procedure

FIG. 1 shows an example GUI for the initial typing characteristics measurement stage. A block of text is dynamically generated and displayed to the user (110). The user is prompted to type the displayed text into a text field (120). While the user is typing, the system records the time the user presses and releases each key. Once the user has completed typing the dynamically

generated text, the users typing information is then passed on to the neural network algorithm described in FIG. 2.

FIG. 2 shows the process through which the neural network is trained for each new user. This algorithm executes when the user is creating a new account. The algorithm generates a data vector, which is a vector representing the users typing characteristics, from the typing data, and a weight vector, which is used by the neural network, from random values (240). The data vector contains information that is vital about the users typing characteristics, including but not limited to time of depression of each key and time elapsed between each keystroke. The weight vector contains values which represent the weights of each node in the neural network. The neural network is made up of multiple layers of nodes which each have given inputs, weights, and outputs. The first layer of the network contains a node for every element in the data vector. The last layer of the element contains only one node, which outputs the final result. Hidden nodes in the middle layers provide an intermediary between the first and last layers. Each node takes its given inputs from the previous layer (or from the data vector, as is the case with the first layer), performs a mathematical function on this data using the nodes values and weights, and then outputs the final result to the next nodes (or as the final output of the program, as is the case with the last layer). If the output is above the threshold for acceptable values, it will be treated as an acceptable output for authentication, and if it is not then it will not be sufficient for authentication. The neural network must be trained when the user first creates an account (5 and 6). The program runs data vectors to which it already knows the final result (e.g. the data vector generated from the users typing, which has a desired output value of 1 (or success), and data vectors stored in the database generated from other users typing, which have desired value 0 (or failure)) through the network, and adjusts the weights to achieve the desired result until the network returns the optimal result (i.e. changing the weight vector does not improve the results to a measurable extent). If the value returned with the new users data vector does not meet the threshold for acceptable values, the training process is repeated with a new set of randomized weight vectors. Once an optimal weight vector is created, it is stored in the database under that user.

FIG. 3 describes the authentication algorithm which takes to authenticate the user. The algorithm is the same whether the system is continuously monitoring typing patterns during program use or using a one-time measurement of typing patterns at log-in time, with the only difference being

the log-in time algorithm generates text for the user to type at step 320. The user is first authenticated via their username and password with which they created their account on this system (310). Then the program measures the user's typing patterns (320), either after a set amount of time for continuous authentication or after prompting the user to type dynamically generated text for log-in authentication. The program measures the user's typing characteristics by recording the times when keys on the keyboard are depressed or released. This data is then used to compute statistics which describe the user's typing characteristics, e.g. the user depresses the 'A' key for a measured average of 80 milliseconds, or the user takes a measured average of 200 milliseconds between releasing the 'A' key and pressing the 'B' key. The algorithm generates a data vector, which is a vector composed of the aforementioned statistics and representing the users typing characteristics, from the typing data (340). The data vector is then run through the neural network (350) to return a final output between 0 and 1. If the output meets the threshold for acceptable values for authentication (360), then the user will be granted access to the system; otherwise, the user will be denied access.

5 Testing and Results

Testing was conducted to determine and optimize the accuracy and breach rate of this neural network approach. By breach rate, I mean the percentage of people other than the original user to whom the network would incorrectly grant access. To do this, a program was created which prompted users to type in a given amount of random text chosen from a large selection of text, measured the users typing characteristics while the user typed this text, and then stored their typing data on a server. This program was posted online at http://www.lukeknepper.com/research.html, and the website was spread via online social networks in order to collect data to test. After two weeks, 1601 sets of valid typing data were collected. These data sets were then used to train the neural networks and to test the accuracy of the neural networks.

Another program was written to automatically run tests on the neural network. The program tests each data set in the sample in the following manner: it trains a neural network using the given data and a small sample of randomly selected data sets from the larger sample, then it runs each data set in the sample through the trained network and records how many breach the

network (i.e. are granted access) at different threshold levels. The program runs through this process thousands of times, covering all of the data files multiple times. This testing found the optimal threshold value to be .007, which had a breach-rate of about 9.5Various improvements were made to the neural network and then analyzed using the same testing technique described above. The first improvement that was tried was increasing the amount of exposure the neural network received to other data files. At training time, the neural network is given one data file for the user so it knows what a correct data set looks like, and is also given many data sets from other users so it knows what incorrect data sets look like. Testing suggested the optimal number of data sets was 25: this was the minimum number to lower the breach rate below 10

Here is a summary of the major tests described for a base network with one layer, 25 data files, and 1000 cycles per file. For more detailed data, see Appendix A.

Network Type: Time —— Breach Rate

Base Network: 1.1 s 90.50Increased exposure (25 ? 50): 2.3 s —— 90.70Increased Cycles (1000 ? 2000): 2.7 s —— 90.90Increasing Layers (1 ? 3): 3.1 s —— 90.50Increasing All: 7.5 s —— 90.90

6 Conclusion

This project shows that neural networks can effectively be used to create a typing characteristic authentication system. While this method, at around 90

6.1 Additional Applications

One additional application of this technique is to offer continuous authentication by constantly monitoring typing characteristics during program use. A program (FIG 6) has been nearly completed to simulate the continuous authentication process in action. This program is a simulation of an instant messaging program, where the user interacts with a pre-programmed bot which randomly asks the user questions to which the user is supposed to type answers. The program measures the user's typing characteristics upon the user's first response, when the user is prompted to type a simple sentence containing every letter of the alphabet, and trains a neural network to this

typing data. The program then measures the user's typing characteristics for every subsequent response and runs these data through the created neural network. If the neural network does not output a sufficient value (i.e. one that is greater than the standardized output) then the warning level is raised. Once the warning level reaches the 100% threshold, the system shuts down and locks the user out of the system. The warning level is lowered with every response which matches the original typing characteristics.

References

[1] Cho, S. and Han, D. "Apparatus for authenticating an individual based on a typing pattern by using a neural network system." http://www.freepatentsonline.com/6151593.html

[2] Maisuria, L. "A COMPARISON OF ARTIFICIAL NEURAL NETWORKS AND CLUSTER ANALYSIS FOR TYPING BIOMETRICS AUTHENTICATION."

[3] Nallusamy, R. and Dr. Duraiswamy, K. "Neural networks for dynamic shortest path routing problems-A survey." 2008.

[4] De Oliveira Paula, Marcus V.S. et. al. "User Authentication based on Human Typing Pattern with Artificial Neural Networks and Support Vector Machine "

[5] Ponnath, Abhilash. "Instantaneously Trained Neural Networks."

[6] Peacock, A. et al. "Typing Patterns: A Key to User Identification." http://www2.computer.org/portal/web/csdl/doi/10.1109/MSP.2004.89

[7] Stomski, Paul J. Jr. and Adel S. Elmaghraby. "SELECTION OF A NEURAL NETWORK FOR VISUAL INSPECTION."