NEURAL NETWORK TYPING AUTHENTICATION

Luke Knepper || lukeknep@stanford.edu
Computer Systems Lab 2009-2010

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 provide an extra level of security to current authentication techniques, helping ensure an intruder does not gain wrongful access to a user's account.

BACKGROUND AND INTRODUCTION

Typing patterns differ by person. People naturally hold down specific keys for specific times and take longer between different keystrokes. These typing characteristics can be, and have been, used for authentication purposes. In this project, I propose and test the accuracy of using typing authentication methods to boost security. Log-in time and continuous typing pattern authentication can help ensure that an intruder does not wrongfully gain access to a user's system, whether by stealing their password or using their account after they have logged in. Previous research that has been found on the accuracy of authenticating by typing patterns concurred on two results: Neural-network algorithms are the optimal approach for this goal, and such methods are on average 80-90% accurate. These findings suggest that typing characteristic authentication can offer a powerful boost to security.

Neural networks are modeled after the human brain. They are composed of Nodes (i.e. neurons) which take inputs from previous nodes, compute a simple function from these Inputs, multiply the result by a given weight and then pass the result on to the next nodes (see Fig. 3 for a simple diagram).

DEVELOPMENT

A log-in simulation program (Fig 1.) 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 algorith 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 (See Fig. 2) was developed, was posted online, and collected 1,600 sample data files.)

FIGURES

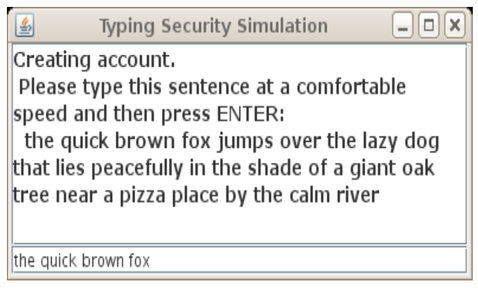

Fig 1: The log-in program

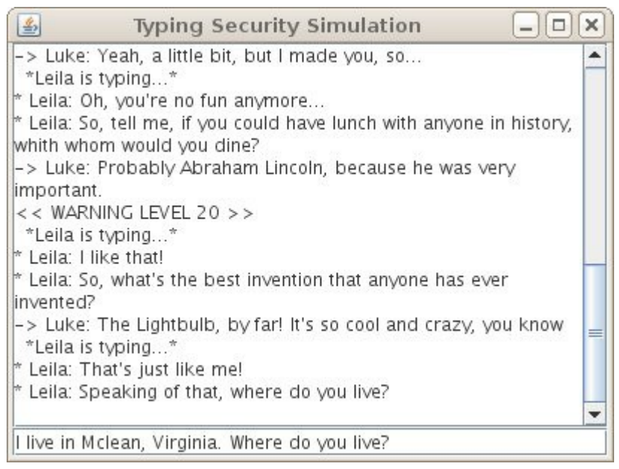

Fig 2: The continous authentication simulation program

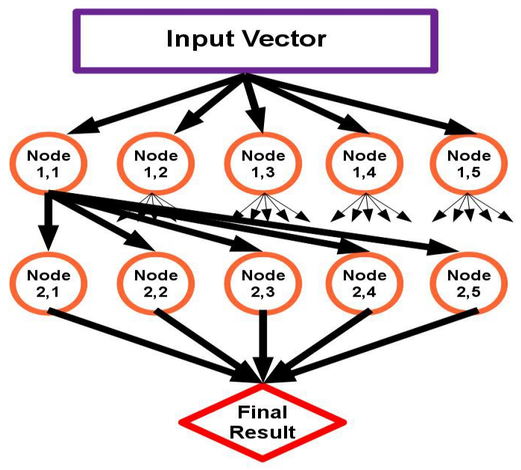

Fig 3: An outline of a simple neural network

Network Type	Mean Training Time	% Breached (threshold of .007)
Base Network	1.1 s	9.5%
Increased exposure (N: 25 → 50)	2.3 s	9.3%
Increased Cycles (C: 1000 → 2000)	2.7 s	9.1%
Increasing Layers (L: 1 → 3)	3.1 s	9.5%
Increasing All	7.5 s	9.1%

Fig 4: A summary of the results of testing

DISCUSSION

In order to test the effectiveness of the neural network structure, the following testing procedure was used:

1. Train a neural network for each data type in the data set.
2. For each network, run every other data file in the data set through that network. Record the returned value.
3. Calculate the average number of data files that breach the network (return a value within the threshold) for a range of different thresholds.

The type of network used was then modified in a variety of ways and the statistics for each type of network are stored in the table above.

The findings suggested that .007 was the optimal threshold value for this algorithm, as under 10% of the files breached at this value, and the user was still able to access their account able 80% of the time.

The minimum network settings for an accuracy of over 90% (i.e. less than 10% breach rate) were found to be a single-layered (L=1) network with 25 randomly selected files used (N=25) and 1000 cycles per file (C=1000). The main modifications tried were increasing the layers (L), number of files (N), and cycles (C). As shown by the table above, the various modifications did not significantly improve the accuracy of the network approach, but did consume significantly more time.

These findings suggest that a simple neural network can provide an extra layer of authentication security. However, at a breach rate of 10%, the neural network typing authentication approach is not nearly effective enough to replace passwords, and should only be used as a security supplement. While it may cause the user a small inconvenience if they have to attempt to log-in multiple times, the security boost is worth the cost.