Proceedings of Student-Faculty Research Day, CSIS, Pace University, May 5th, 2017

Continuous Authentication through Genetic Algorithms and Eigenvector Decomposition

Rony Alvarez Arzu, Siddhesh Andhari, David Douglas, Alexander Mauro, and Gene Locklear
Seidenberg School of CSIS, Pace University, New York, NY

***Abstract*—An eigenvector is a vector whose direction does not change when linear transformation (scaling, sheering, etc.) is applied to it and thus represents the ’characteristic vector’ of a matrix. In this paper, we investigate the possibility of building a classifier that can continuously authenticate users based on the eigenvector decomposition of their associated ’User Matrix’. This is done in part by determining the linear transformations made to the eigenvector(s) of a ’User Matrix’ to produce the other vectors contained in the ’User Matrix’.**

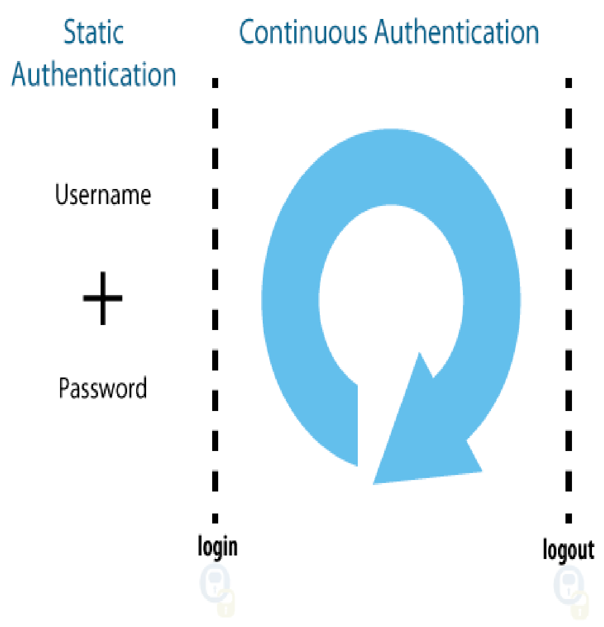
We define an individual’s ’User Matrix’ as 200, 22-dimensional vectors, each of which captures the flight and dwell time of their typing keystroke. We propose a classification system that determines the eigenvector(s) of an individual’s ’User Matrix’ and then applies a series of transformations to an unauthenticated but verified typing vector *UV* in order to determine if *UV* is compatible with the other vectors in the ’User Matrix’. Compatibility, in this sense, is defined as the ability of the transformations made to *UV* to reproduce one of the eigenvectors of the individual’s ’User Matrix’. This can be thought less formally as a system that determines the sequence of transformation made to an eigenvector, of the ’User Matrix’, that created the other vectors in the ’User Matrix’. The system determines this process and reverses it in an attempt to reproduce the unauthenticated vector. If the unauthenticated vector can be reproduced, within some threshold, then the user is authenticated.

I. Introduction

A. The Continuous Authentication System Methodology

Passwords are not ideal because they depend completely on the user to adhere to very complex guidelines on how to create and use secure passwords. Typical robust passwords require the user to create passwords that are lengthy, do not use common words, contain numbers and symbols, and mix letter cases. Unfortunately, because following these rules for creating robust passwords makes the passwords too difficult to remember, users often use *weak* passwords. Thus the user often reverts to the *path of least resistance* when creating passwords by using their own name, telephone number, mothers name etc. [6]. Less obvious but equally poor is a choice of passwords like 1QAZ@WSX which at first appears to be a strong alphanumeric password, but a quick examination of the QWERTY keyboard shows that this is an easily predictable pattern. Additionally, a related problem is the use of too many passwords. Users are often required to have multiple, different passwords and because remembering them is difficult they tend to write them down. While such action is understandable, it can make the user more vulnerable to attack.

Fortunately, the weakness of password-based security has been recognized commonly by experienced system security experts and efforts are well underway to find more robust, alternate solutions but are not intrusive to the user [5]. There are many new technologies which have arisen such as tokens, certificates, etc. designed to supplement the password-based system [9]. Yet none of them fully addresses the central limitation of having only a single check for a user’s authenticity. Unauthorized users thus, once obtaining admittance, may continue for extended periods to gain access to critical information unless the system can be made to routinely re-authenticate the user’s credentials [8]. Simple solutions to re-authenticate such as locking a user’s computer after some period of inactivity or requiring the entrance of a PIN every few minutes are in wide use but are both intrusive and highly annoying to the user [1]. I believe more ideal re-authentication system could be both passive and transparent, interrupting the user only when some anomaly has been detected in their conduct. Continuous authentication addresses this problem of re-authentication by periodically checking to see if the current user is the same one who was originally authenticated. Continuous authentication passively tracks user activity using some relevant sensor and comparability metric and interferes only when a more-established threshold has been crossed [7]. Nonetheless, the idea of continuous authentication is not new and methods are already being tested that may prove effective [11]. These methods center on human physical biometrics and include facial recognition, fingerprint scanning, and multi-modal physiological sensors [1]. However, the implementation of continuous authenticating security systems is not trivial. These systems must be able to authenticate the user without his active cooperation, not overwhelm the system resources and be highly reliable. Even more importantly, the sensors that the system relies on must be able to consistently capture some universally available trait possessed by any user and be immensely discriminable.



B. *Continuous Authentication and Biometrics*

Biometrics is a very general term and might be used to describe either a physical or behavioral characteristic or the method for recognizing such characteristics [3]. The most typical biometric systems are composed of five integrated components [2]. These components consist of (1) a sensor that is used to collect and digitize the users data, (2) a signal processing algorithm which creates a usable user template, (3) a data storage component which stores the data which will be measured against the user template,(4) a matching algorithm which does the comparison of user templates to new templates in the data storage, and (5) a decision process which uses the information provided by the matching algorithm to make the required decision for the user. In recent years, work in biometrics produced an alternative to passwords [8]. Biometrics, unlike passwords, rely on authenticating an individual based on some physiological or behavioral characteristics. These characteristics are an integral and unchangeable part of the individual and do not require the user to remember anything. Beyond the physical characteristics, biometrics also includes behavioral and cognitive traits, such as how the user types on a keyboard or chooses words in sentence construction [10]. The way an individual types or the word choice she uses in sentence construction are an inherent part of her mechanical and cognitive skills and cannot easily be changed [11]. Thus biometrics provides not only an improved way to authenticate but a *much improved way* in which to perform continuous authentication.

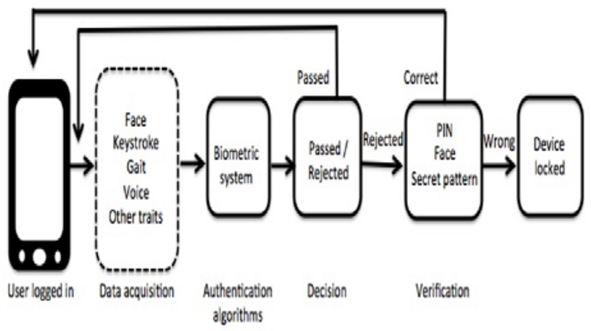
C. *Keystrokes as a Biometric for Continuous Authentication*

Keystroke dynamics is the detailed timing information that describes exactly when each key was pressed and released as a user types on a keyboard [3]. Keystroke dynamics is concerned with behavioral biometric contained within the rhythms a user follows while typing [5]. These rhythms give rise to a unique and distinguishable pattern for individual users which can be used in an authentication system [6]. Two characteristics are used to define keystroke dynamics are *Dwell Time* (the time between the instant a user presses a key and the instant the key is released) and *Flight Time* (the time between the instant a user releases a key and the instant he presses the next key) [7]. The dwell time and flight time are captured by keystroke logging, which is done by a key logger that records and timestamps key presses. Because keystroke dynamics is a behavioral biometric, it is considered by some not to be as reliable as a physical biometric [cite]. Behavioral biometrics use a probabilistic method for determining a match while physical biometric use a binary method (pass/fail). Matching success is determined through measuring false acceptance rates and false rejection rates, which, because of the probabilistic nature of keystroke dynamics, is not a linear relationship [4]. Nonetheless, keystroke dynamics have very beneficial aspects when used in continuous authentication. [4].

D. *Continuous Authentication on Mobile Devices*

Continuous Authentication on smartphones was developed to counter the potential attacks that can happen after a smartphone has been unlocked [11]. Smartphones are the standard personal device that many of us use to store sensitive information. They are a widely used gadget and are very vulnerable to loss or theft. Unfortunately, the vast amount of personal data stored on these devices has made them a target for criminals. Continuous authentication was proposed to prevent this information from being retrieved by criminals [8].

Hugo Gascon, Sebastian Uellenbeck, Christopher Wolf and Konrad Rieck [1] presented a user authentication approach that is based on analyzing the typing behavior of a specific user. The approach was developed to counter attacks that happen only after the device has been unlocked [10]. Theirs approach uses a continuous authentication scheme by means of biometry while the user is entering text [3].



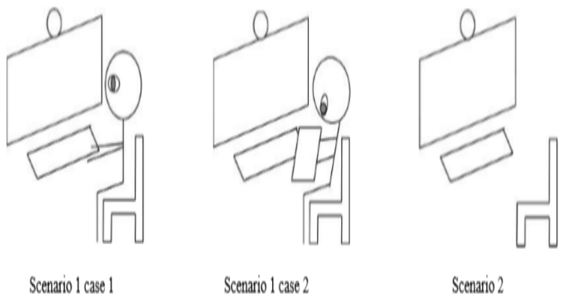
E. *Authentication using Temporal Information*

Most systems nowadays only authenticate users at the initial log in session. As a result, it is possible for an attacker to access all the user's information and resources with or without the permission of the owner until he or she logs out.

Continuous authentication uses temporal information color information of the user's clothing in addition to information about user's face. They system uses the webcam to automatically register both clothing color and face information every time the user logs in [7].

Koichiro Niinuma and Anil K. Jain [2] proposed a user authentication approach that is based on this concept. Their approach uses the web cam to register the clothing and facial information of the user when he or she log in, and keeps checking this information and does not require re-authentication while the user is in front of the computer. The system requires re-authentication every time the user walks out.

Their system distinguishes between two scenarios. The first scenario is divided in two cases. When the user is sitting in front of the machine and is facing the webcam, and when the user is sitting in front of the machine but is looking down. The second scenario is when the user has moved away from the machine.



F. *System Requirements*

The concept of continuous authentication is not new, and there is no doubt to security professionals that it is a very important one. However, continuous authentication has entered mainstream computing in a meaningful way yet. Nevertheless, with the amount of private information that we store in our personal devices, it is a necessity to protect them at all times. For example, the average user walks away from the computer for short breaks without logging out of the system [9]. This is an opportunity for attackers to access the system and all of the user's information.

G. *Authentication Process and the User Matrix*

We propose the following for our continuous authentication system:

H. Use – The system will continue to authenticate the user in the background as long as the user is actively using his keyboard. It will not ask the user for re-authentication, even if the user walks away from the computer.

I. Cost - Cost is a very important factor for most people. For this reason, the authentication system should only use the standard devices (e.g. keyboard) and avoid the use of any external or special type of device.

J. Security – The system will be always active in the background and will not require the users to re-authenticate every time they walk away from the computer.

K. *Eigenvectors and the User Matrix*

An eigenvector v is known as the characteristic vector of a matrix and v is a non-zero vector that does not change direction when it undergoes a linear transformation. We can imagine that the 'User Matrix' M_{User} contains the vector v and that this vector v is an eigenvector of the M_{User} if there exist some transformation of v say $T_{Linear}(v)$ that can be represented as a scalar multiple of v. Thus $T_{Linear}(v) = \lambda v$ in which λ is a scalar. Applying $T_{Linear}(v)$ only scales v by some scalar value λ and thus is known as an eigenvalue of *MUser*.

L. Eigenvector Decomposition

The major drawback to utilizing eigenvectors in our classification scheme is that the M_{User} is not square (i.e. 200 x 22). In order to determine the eigenvectors, we must use *Singular Value Decomposition* (SVD).

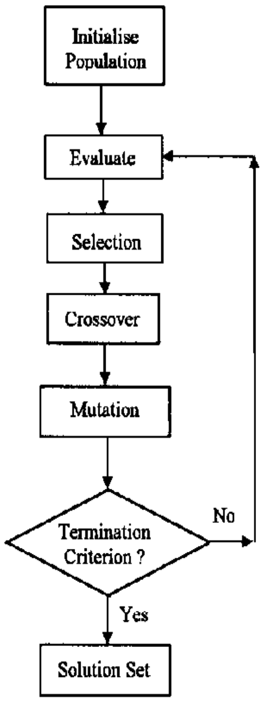
SVD of the M_{User} is a factorization of the matrix into a unitary matrix U, a diagonal matrix Σ, and a unitary matrix V^T. The column vectors of the matrix U are known as *Left-Singular Vectors* and are a set of orthonormal (mutually orthogonal and unit length) eigenvectors created by $M_{User}M^T_{User}$. The column vectors of the matrix V are known as the *Right-Singular Vectors* of M_{User} and are a set of orthonormal eigenvectors created by $M^T_{User}M_{User}$ The *Singular Values* are the diagonal entries of Σ which are the square roots of the eigenvalues of both U and V.

M. Use of Genetic Algorithms

The basis for our proposed classifier is the determination of the linear transformation(s) made to the eigenvector(s) of M_{User} such that the unknown vector v_{unk} can be replicated to within some threshold. This can be stated as $T_{Linear}(v) \approx v_{unk}$ where we do not know the nature of T_{Linear}.

While we do not know the nature of $T_{Linear}(v)$, we do know the possible operations that can be associated with it. These operations are scale, translate, rotate, reflect and/or shear. Thus, we know that in some sequences, a combination of these operations will allow us to transform the eigenvector v into the unknown vector v_{unk} to within preset tolerance.

The determination of the required operations and their proper sequence takes the form of an optimization problem and is ideally suited for a possible solution by the use of genetic algorithms. In all genetic algorithms, a *population* of possible solutions is created and then *evolved* toward the best or better solution. In our case, each member of the population (i.e. solution space) has n properties that are the possible linear transformations (scaling, shearing, rotating etc.). These operations can be included or excluded from each possible solution and their sequence within the solution can also be changed. The goal of using genetic algorithms is to determine the correct linear transformation sequence from within a search space that is theoretically infinite.



Biometric authentication systems are steadily becoming a solution to wide number of authentication and identity management problems [11]. Unique parts of human body that can be recognized and used as a mean to identify a person include fingerprints, iris, lips, etc. [13]. Though fingerprint and facial recognition systems are most widely used and developed, most of the systems and methods are slow or require expensive technical equipment [2].

In the cyber security field, Genetic Algorithm (GA) is an invaluable tool for solving optimization problems due to its robustness. It does not break even in the presence of a reasonable noise or even if the inputs are changed slightly. GA offers significant benefits over other optimization techniques in searching a large state space or n-dimensional surface. In today's information age information transfer and sharing has increased exponentially. With the popularization of Internet and exponential increase in e-commerce transactions security has become an inevitable and an integral part of any e-commerce application. Data integrity, confidentiality, authenticity, non-repudiation has gained tremendous importance and have become important components of information security. In this paper we have made an attempt to exploit the randomness involved in crossover and mutation processes of GA for generating a barcode for authentication process. The number of crossover points and number of mutation points is fixed and cannot be altered by the user. In the current work we have employed a single crossover point and two mutation points. We can use Code-39 and Code-128 encoding techniques for generating a barcode. The barcode data comprises of 12 randomly generated decimal digits. Each decimal digit is represented using 4 bits. Hence the length of the barcode data is 36 bits. The randomly generated data is transformed into encoded form by applying crossover, mutation and XOR operations before generating a bar code. The randomness together with encoding makes the password robust and hard to track. Finally, the algorithm is implemented in Java and applied for authentication of employee data in a hypothetical organization. The methodology is general and can be applied to any task where authentication is required [5].

A. Keywords: Genetic Algorithm, Cross-over, Mutation, Barcode, Encoding.

Genetic Algorithm

Generally, a Genetic Algorithm consists of three basic operations.

• Selection

• Crossover

• Mutation

The first step consists of searching individuals for reproduction. We have generated a genetic pool consisting of 50 twelve digit numbers representing the chromosomes which are randomly generated and from which a single random number with the highest fitness value as dictated by the fitness function is selected. The random number thus selected is divided into two parts and encoded using cross-over and mutation operations before generating a code using code-39 encoding technique.

Cross-over is the process of taking two parents and producing from them a child. In an optimization problem, crossover operator is applied to the mating pool with the hope that it creates a better offspring. For the problem under consideration, crossover is taken as one of the steps in producing a decrypted vector. We have employed four-point crossover method. In the case of optimization problem, selecting more than four crossover points will result in the disruption of building blocks whereas in the case of encryption larger the disruption better is the algorithm which makes it robust and difficult to break.

After crossover, the vectors are subject to mutation. In optimization problem, mutation prevents the algorithm from being trapped in a local minimum. Mutation plays an important role in the recovery of the lost genetic matter as well for randomly distributing the genetic information. In encryption problem, mutation is employed for inducing disorder into the vector. It introduces a new genetic structure in the population by randomly modifying some of the building blocks and maintains diversity into the population. We have employed flipping method, in which for a character 1 in mutation chromosome, the corresponding character b in the parent chromosome is flipped from b to (9-b) and corresponding child chromosome is produced. In the following example 1 occurs at two random places of mutation chromosome, the corresponding characters in parent chromosomes are flipped and the child chromosomes are generated.

Barcodes consists of a series of lines that vary in width. They correspond to various numeric, alphanumeric, or multi-code configurations readable by a laser barcode scanner. Code 128 is a very effective, high-density zymology which enables the encoding of alphanumeric data. It includes verification protection both through a checksum digit and byte parity checking. This symbology has been widely implemented in many applications where a large amount of data must be encoded in a relatively small amount of space. A Code 128 barcode consists of a leading "quiet zone", one of three start codes, the data itself, a check character, a stop character, and a trailing quiet zone as shown in Fig. 1. The Code 128 data is encoded in strips of bars and spaces. The sequences of zeros or ones simply appear as thicker bars or spaces. The checksum is included in the barcode, and is a digit that verifies that the data just read in was correct. The checksum digit is based on a modulo 103 calculation based on the weighted sum of the values of each of the digits in the message that is being encoded, including the start character.

Pseudocode

```
Step 1: Generate a pool of 50 chromosomes consisting of twelve digit numbers which are randomly generated.
Step 2: Select a single chromosome with the highest fitness value as dictated by the fitness function F given by
        $F = \sum_{i=1}^{12} |d_i - d_{i-1}| + [12 - \mathrm{Max}(r_j)]$ for $0 <= j <= 9$
    where, $r_j$ refers to repetition of digit j and $|d_i - d_{i-1}|$ is the absolute numeric distance between the two digits. Store a selected twelve-digit random number it in a vector.
Step 3: Each decimal digit in step 2 can be represented using 4 binary digits. Hence the total number of binary digits required to represent the data is 4 x 12 = 48 bits. Generate a hash H, by repeating digits 0 and 1 (if the digit is > 8) and 0 and 0, otherwise, required number of times. The hash function generated is such that it enables one-to-one mapping between datasets involved. This renders the hash function reversible.
Step 4: Perform the XOR operation between the data and a 48-bit hash computed above.
```

Step 5: Split the vector into two vectors of size six each.

Step 6: Compute 10's complement of each digit.

Step 7: Perform the crossover operation at the midpoint.

Step 8: Perform the mutation at the extreme positions of the vector. The mutation operation consists of flipping the digit from its original value to its complement.

Step 9: Combine the vectors to reconstruct a 12-digit vector.

Step 10: Perform the XOR operation between the data and a 48-bit hash computed above.

Step 11: Use the 12-digit number generated above to generate a barcode in code-128 format.

Step 12: End

II. Related Work

There are several related works that are used to show the effectiveness of using keystroke biometrics for continuous authentication. Listed below are a summary description and categorization of these works, which we view as similar to our own.

Kilhourhy and Maxion, in 2012, [5] used 20 subjects to compare the ability to classify users using free composition or transcribed text. The work was designed to determine whether free and transcribed text returned equivalent results. They used a statistical classifier (Mahalanobis distance) and a disorder based classifier (Bergadano et al. 2002) on 2 samples from each of 20 users. The experiments determined that the transcription hold times are 2-3 milliseconds slower than free text features, but do not significantly affect the evaluation result. Finally, they believed that, considering the difficulty in collecting free text, it is more appropriate and just as beneficial to use transcribed text.

Acharya et al., in 2012, [6] used keystroke dwell time, keystroke interval time, and mouse movements, from 10 users, in a simulated work environment to determine their suitability as biometric identifiers. Their work was centered on the belief that, due to the unconstrained nature of human-computer interaction, a single biometric is usually not sufficiently robust to determine the user's identity. It was also their belief that a multi-modality multi-biometric model provides the best choice for continuous authentication. They used Naive Bayes classifiers for mapping from feature space to decision space then applied a decentralized parallel binary decision fusion scheme to integrate a set of local binary decisions to a global binary decision. Their conclusions determined a hierarchy of sensor importance that could be used in determining groups of sensors to be implemented in user authentication.

Ahmed and Traore, in 2013, [7] conducted an evaluation involving 53 participants, in an uncontrolled setting, using various typing-based applications. Their hypothesis was that accurate recognition of free text keystroke dynamics is challenging due to the unstructured and sparse nature of the data and its underlying variability and so a new approach is needed. The new approach proposed was using monograph/digraph approximations along with neural network analysis. Neural networks modeled the user behavior, centered on monographs and digraphs signatures of the user. Although the neural network architecture remains the same for all users, the individual user's weights were specific. The experiments yielded a FAR of 0.0152% and FRR of 4.82%, with an EER of 2.46%.

III. Data and Data Analysis

Our choice of data is publicly available through IEEE [8]. This data set represents individual typing vectors of 31 dimensions that consist of key-down presses and key-up releases. Combined together these presses and releases represent the dwell and flight times of each user while they transcribe a specified text string.

B. Data Set

We utilize the Killourhy and Maxion data set [9] which is considered the bench mark data set for keystroke dynamics testing by security professionals. The data set consists of 51 users with a total of 400 samples per user. Each user repeatedly typed a fixed character sequence. tie5Roanl 50 times and completed this typing sequence eight times over eight days. Users were both left and right-handed, male and female, and ranged in age from 31 to 70. The average session time for each user (type 50 passwords) was three minutes.

```
Algorithm for fitness function(i,j)
{
Int fit =0;
While(all nodes at 48.28 covered)
{
If(at distance of 48.28 exist client)
Fit++;
}
Return fit;
}
```

IV. Classification System Framework

Phase 1: Criteria for Solution

In particular, in the fields of genetic programming and genetic algorithms, each design solution is commonly represented as a string of numbers (referred to as a chromosome). After each round of testing, or simulation, the idea is to delete the ’n’ worst design solutions, and to breed ’n’ new ones from the best design solutions. Each design solution, therefore, needs to be awarded a figure of merit, to indicate how close it came to meeting the overall specification, and this is generated by applying the fitness function to the test, or simulation, results obtained from that solution. For the particular instance, the number of client a node can cover is the figure of merit i.e. fitness function. The clients which are already covered are not included in this count.

Phase 2: Define Fitness function

A fitness function is a particular type of objective function that is used to summarize, as a single figure of merit, how close a given design solution is to achieving the set aims.

Phase 3: Choose a Solution

Algorithm:

```
Int Matrix [] [];

 For(i=0;i<number_of_rows;i++)
{
For(j=0;j<number of columns;j++)
{
If client exist
{clients [i] [j]=1;}
Else
{clients [i] [j]=0;}
 }
}

For(i=0;i<number_of_rows;i++)
 {
For(j=0;j<number of columns;j++)
{
If(clients [i] [j]==1)
{
Move 48.28
Fit=fitness(newi,newj);
   If(fit is maximum)
   Clients [i] [j] =2;
   }
```

Phase 4: Apply Solution

Phase 5: Classify

VI. Conclusion

In recent years, several solutions have been proposed to secure our devices to protect our privacy and information. As the continuous interaction of the users with their devices allows for continuous monitoring and analysis of their behavior, we have tried in this paper to retrieve that information and implement a system that will identify a unique typing motion of a user.

Many strategies come in mind when it comes to protecting our devices, for example, the many different types of authentication. The idea of continuous authentication is not new, but it still has a lot of potential and it will only become more relevant with time. The results proposed in this paper open the door for future improvements.

References

[1] ”Identity Theft Resource Center, Non-Profit Organization.” Identity Theft Resource Center. Web. 17 Jan. 2014.

[2] ”Ethical Hacking Tools and Techniques: Password Cracking.” Password Cracking Tools and Techniques. Web. 11 Jan. 2014.

[3] ”PPA Help.” PPA Help. Web. 7 Jan. 2014.

[4] ”Top Ten Password Cracking Techniques.” Technology, News and Reviews. Web. 7 Jan. 2014.

[5] ”The Keys to Continuous Authentication.” - BankInfoSecurity. Web. 10 Jan. 2014.

[6] ”How Behavioral Biometrics Will Transform Network and IT Security.” Plurilock. Web. 12 Jan. 2014.

[7] ”How Do Biometric Systems Work?” - BSI Shop. Web. 12 Jan. 2014.

[8] Hoang, Bichelien, and Ashley Caudill. ”IEEE - Biometrics.” IEEE.org. Emerging Technologies Portal, 2006. Web. 10 Mar. 2014.

[9] S. Mondal and P Bourns Continuous Authentication Using Behavioral Biometrics Norwegian Information Security Library, Web. 10 Mar. 2014.

[10] David. E. Goldberg “Genetic Algorithms in Search Optimization and Machine Learning” Pearson Education 1989 ISBN-13: 978-020115767.

[11] X. F. Liao, S. Y.Lai and Q. Zhou. Signal Processing. 90 (2010) 2714 – 2722.

[12] Dr. Poornima G. Naik, Mr. Girish R. Naik “Secure Barcode Authentication using Genetic Algorithm” www.iosrjournals.org e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 2, Ver. XII (Mar-Apr. 2014), PP 134-142

[13] Hugo Gascon, Sebastian Uellenbeck, Christopher Wolf, and Konrad Rieck. Continuous Authentication on Mobile Devices by Analysis of Typing Motion Behavior.

[14] Koichiro Niinuma and Anil K. Jain. Continuous User Authentication Using Temporal Information. Fujitsu Laboratories