Identifying Users with Application-Specific Command Streams

Alaa El Masri, Harry Wechsler
Department of Computer Science
George Mason University
Fairfax, VA, USA

Peter Likarish
Department of Mathematics and Computer Science
Drew University
Madison, NJ, USA

Brent ByungHoon Kang
Graduate School of Information Security
Korea Advanced Institute of Science and Technology
Daejeon, South Korea

Abstract*—This paper proposes and describes an active authentication model based on user profiles built from user-issued commands when interacting with GUI-based application. Previous behavioral models derived from user issued commands were limited to analyzing the user's interaction with the *Nix (Linux or Unix) command shell program. Human-computer interaction (HCI) research has explored the idea of building users profiles based on their behavioral patterns when interacting with such graphical interfaces. It did so by analyzing the user's keystroke and/or mouse dynamics. However, none had explored the idea of creating profiles by capturing users' usage characteristics when interacting with a specific application beyond how a user strikes the keyboard or moves the mouse across the screen. We obtain and utilize a dataset of user command streams collected from working with Microsoft (MS) Word to serve as a test bed. User profiles are first built using MS Word commands and identification takes place using machine learning algorithms. Best performance in terms of both accuracy and Area under the Curve (AUC) for Receiver Operating Characteristic (ROC) curve is reported using Random Forests (RF) and AdaBoost with random forests.

***Keywords*—Active Authentication; Behavioral biometrics; Intrusion Detection; Machine Learning**

I. INTRODUCTION

While multifactor and multimodal authentication models provide robust user identification mechanisms, concerns of verifying the user's identity beyond initial authentication have dominated recent discussion. The problem of ensuring that the active user is still the same user who was initially authenticated led researchers to search for additional means of identification that span the duration of the user's active session and provide continuous authentication. This concept, sometimes referred to as active or continuous authentication, requires that a system attempts to certify that the identity of the authenticated user stays the same throughout his/her active session. Behavioral biometrics serve as good candidates for such systems because unlike other authentication schemes user's characteristics can be observed and collected discretely without the user's awareness that verification takes place.

Despite the wide consensus on the vulnerability of password-only authentication, 52% of organizations still employ such policies [1]. There has been a push for organizations to strengthen their authentication policies by adopting two-, and sometimes three-factor authentication, in which the user is required to provide additional means of identification. Biometric-based authentication can be leveraged as a primary or additional factor in multi-factor authentication.

Biometric-based authentication is divided into two classes: physiological and behavioral biometrics. Physiological biometrics (i.e., biological biometrics) are based on a person's physical characteristics such as fingerprint, face, and iris, among others. Behavioral biometrics are based on a person's behavior such as keystroke dynamics [2], gait [3], or command line lexicons [4]. It is important to note that the various biometrics techniques are not always designed and implemented independently as recently, researchers [5]–[8] have explored fusing biometric identifiers to create a multimodal biometric system and improve the identification process.

This paper proposes and describes an active authentication model based on behavioral biometrics pertaining to GUI-based application user-issued commands. Rresearchers have explored the idea of building users profiles based on users' behavioral patterns when interacting with such graphical interfaces, especially in the area of human-computer interaction. They did so by analyzing the user's keystroke and/or mouse dynamics. However, most of the work that focused on creating behavioral model from user issued commands, and perhaps the most relevant to our work, were limited to users' interaction with the *Nix command shell program [9]–[12]. Never before the idea of creating profiles by capturing users usage characteristics when interacting with a specific application's GUI has been examined, which goes beyond how a user strikes the keyboard or moves the mouse across the screen. It provides more dimensions to consider and a richer set of behavioral features to include when building behavioral profiles.

To answer the question of whether or not a stream of commands triggered by users' interaction with a GUI-based application can serve as a behavioral biometric, we have chosen to utilize and repurpose a dataset collected by the MITRE Corporation from previous research on organization-wide learning and recommender systems [13]. This dataset represent two years of users' usage data from interacting with Microsoft (MS) Word (see section III). The motivation behind our consideration of the MITRE dataset is due to the following:

- MS Word is a more popular medium among the common computer user than the *Nix command shell program. Microsoft claims that roughly half a billion people use MS Office [14]. Furthermore, a recent report by Forrester shows that 84% of organizations are using MS Office 2010 as their office productivity suite [15]. Therefore, creating user behavioral profiles from interacting with such widely used applications would have a broader applicability.
- The graphical user interface of MS Word provides a richer set of behavioral features when compared to the command line interface (CLI) of the Linux shell. The possible feature set is unique to the application in use and can be derived from the way a user interacts with its properties. While the current dataset is strictly a representation of the commands issue by their users, a future consideration would explore a richer set of behavioral features such as the average velocity of scrolls, the ratio of backward and forward scrolling switches and the average elapsed time between scrolls, to list a few. On the other hand, the only interactions a user has with the Linux shell are confined to the commands it accepts.
- The last, and most obvious, reason is the availability of the dataset from a previous work by MITRE that is made public.

The contributions of our paper come in method, performance, validation, and venues for future application-specific behavioral biometric research. The method combines behavioral biometrics for user profile representation with learning methods suitable to discriminate among them. Best AUC (Area under the Curve) performance, when user profiles are modeled in terms of MS Word commands and validation takes place using k-fold cross-validation, is obtained using Random Forests (RF) as the learning method of choice. Compared to other learning methods used, RF performs much better due to its intrinsic characteristics, e.g., randomness and sampling, and ensemble methods, which cope better with the varying nature of individual user profiles. That is, RF trains multiple "weak" classifiers, in our case Decision Trees, based on a randomly selected subset of the entire feature space. RF then makes the classification decision based on the majority vote of these weak learners. The relevance for AUC comes from its relation to how thresholds are set and used when tracing the ROC (Receiver Operating Characteristic) curve.

The outline of this paper is as follows. Section II provides a background on the concept of active authentication and a review of related work. Section III describes the data used to test our active authentication scheme. Section IV describes the methodology proposed and the tools that have been implemented to support it. Section V reports experimental results and highlights the potential of our approach. Finally, section VI concludes our work with a summary of contributions, impact on security, and venues for future research and development.

II. Background and Related Work

The concept and the use of continuous or active authentication has gained momentum in recent years for the purpose of verifying the identity of a user beyond initial authentication including interest from the defense sector [16]. Traditional authentication models were designed to authenticate users at the initial stage and, consequently, grant or deny access. However, once a user is granted access there is no way to further attest the authenticity of the user and find out illicit use, if any. A continuous authentication system attempts to address this flaw by re-authenticating the user throughout his/her active session. It is important to point out that much of the work in the field of intrusion detection has also utilized behavioral biometrics, and thus lends itself to active authentication. Both active authentication and intrusion detection are about finding imposters engaged in illicit use of resources. The following review of the related work covers both concepts.

Various implementations of continuous authentication systems have been proposed and all have, naturally, employed biometrics as the authentication type. While the related work discussed here is not an exhaustive review of the evolution of biometrics, it does highlight, at various points in time, the research and technologies applied in this field. For example, Klosterman and Ganger [17] used a camera mounted on top of a computer monitor to capture the user's face images to determine identity. Azzini et al. [8] proposed and tested a multimodal authentication system that utilizes two physiological traits, a fingerprint and face biometrics. The user is initially authenticated by providing his/her username and password. A corresponding template is then retrieved for the face recognition matching phase, which continuously checks the identity of the authenticated user. Once the face biometrics module become unsure of the user's identity, the user is prompted to provide his/her fingerprint for re-authentication. Yap et al. [18], on the other hand, designed a system that utilizes a camera and a biometric mouse that continuously collects both the face and the fingerprint features and fuses them into a composite score that is checked against a threshold to determine authenticity.

While the above examples rely on external sensors for collecting the biometric data, others have resorted to utilize more traditional input tools such as the keyboard and mouse. For example, many have investigated users' keyboard typing characteristics (i.e., keyboard dynamics) as possible means of continuous authentication. A survey of such method is provided by Shanmugapriya et al. [19], which discusses the various work in this area and compares the different keystroke metrics used. Others [20]–[22] have explored the idea of mouse dynamics as a possible user biometrics. A survey by Revett et al. [23] provides a detailed review of work in this area and proposes a new one. Ahmed and Traore [24] have combined both keystroke and mouse dynamics for intrusion detection purposes. Others have fused such techniques to create multimodal biometric systems. For example, Grag et al. [25] architect a framework for active authentication that depends on keyboard activity, mouse movement coordinates, mouse clicks, system background processes and user run commands. Unlike our work, their techniques depend on features collected from the users' interactions with the GUI-based operating system in general and not application-specific features, which is more granular in scope and can be geared towards critical applications that deal with classified material. Such active authentication technique allows the monitoring of a richer set of behavioral features that reveal a cognitive process that may help to infer a person's knowledge, intentions and, more importantly, identity.

Another popular means of continuous authentication is the analysis of command line lexicons, which is closely related to our approach. For example, Lane and Brodley [9] used the Linux command line prompt to capture a sequence of fixed-length commands and create users' profiles. Detection of normal or abnormal behavior is then calculated by measuring the similarity of new sequence to existing ones.

Marin et al. [10] have also used command lines to detect intruders. They classify legitimate users into categories based on the percentage of commands they issue in a given time period. A total of 5,000 commands per user were collected from a Linux shell for 50 different users. Users' profiles were created from groups of 1,000 commands and, therefore, each user was represented five times. Their classification methodology employed expert rule to reduce dimensionality, K-means for clustering, a generic algorithm for further dimensionality reduction, and Learning Vector Quantization (LVQ) for refining clusters. Their work has resulted in a classification rate close to 80% and misclassification rate less than 20%.

Yeung and Ding [11] have used user profiles created from Unix shell commands to apply dynamic and static behavioral modeling approaches. The difference between the two approaches is that dynamic models explicitly model temporal variations. The dynamic model is based on hidden Markov model (HMM), while the static model is based on event occurrence frequency. They applied novelty detection instead of classification due to the lack of data representing abnormal behavior for training. Their experiments showed that the static modeling approach yielded better result than the dynamic modeling, achieving at its best a true detection rate (TDR) of 87% and a false detection rate (FDR) of 20%.

Schonlau et al. [12] also used Linux shell commands and applied six different classification methods for detecting masqueraders – people who impersonate the authenticated user. Over several months, 15,000 commands per user from 70 users were obtained. 50 users were selected to be intrusion targets and the remaining 20 user were designated as the masqueraders. Data from the masqueraders was interspersed into the targets data. The first 5,000 commands are used to train the user profiles. For each block of 100 commands, a score is computed and checked against a threshold. Out of the six methods applied the Bayes 1-step Markov performed the best in detecting masqueraders but failed in achieving the desired goal of 1% false alarm rate (FAR). On the other hand, the Uniqueness method (based on command frequency) did well in coming close to the desired FAR but failed to detect masqueraders. All the other methods performed somewhere in between.

Maxion and Townsend [26] have extended the work by Schonlau et al. by revising the experimental method and applying Naïve Bayes, which resulted in a 56% improvement and 1.3% FAR. More recently the same data set used by Schonlau et al. has been used by Traore et al. [4], which used sequential sampling techniques and naïve Bayes classification scheme and yielded 4.28% false acceptance rate (FAR) and 12% false rejection rate (FRR).

Implementations of command line biometrics have, thus far, been limited to the CLI. In this paper, we explain how such concept can be extended to other domains and in particular to GUI-based applications that provides the user with more than just command-like interactions. However, in this paper we only consider command streams issued by users when interacting with MS Word as means of identification. We test our hypothesis with a data set obtained by MITRE Corporation for their earlier work on a recommender system. A description of this data is provided in section III.

III. Dataset

The data used in this paper was collected by the MITRE Corporation over a period of two year from 1997 to 1998. MITRE monitored commands entered in Microsoft Word by 24 employees. The employees consisted of artificial intelligence researchers as well as technical and support staff. All individuals used the Macintosh operating system. As the system became more robust, additional users were monitored. If an individual switched to a PC, they were dropped from observation. The data was initially collected to explore the development of recommender systems that leverage knowledge of how the entire group used Microsoft Word to tailor the application to the organization [13]. The dataset is publicly available and hosted at http://www.research.rutgers.edu/~sofmac/ml4um/.

Each time a user opened Microsoft Word and executed a command during the period of observation the command was logged along with a unique user ID, the version of Word they used, the file size, the creation date of the file, the operating system and version and the date/time the command was executed. Commands consist of Microsoft Word options, such as “Copy”, “Paste”, “New”, or “Italic”. Over the course of the study, a total of 74,783 commands were observed. The users executed 174 unique commands, and on average each user executed 1,583 commands. Users whom participated in less than 10 sessions where discarded from the data analysis. Table I shows the data characteristics.

TABLE I. Data characteristics

Stat	Value
Total number of commands	74,783
Total number of unique commands	174
Average number of commands per user	1,583
Total number of sessions	11,334
Average number of commands per session	6.57

Stat	Value
Average number of sessions per user	539
Longest session by number of commands	93
Shortest session by number of commands	1

IV. Methodology

This work repurposes the MITRE data to answer the question of whether or not knowledge of application-specific commands can serve as a behavioral biometric. To do this, a profile is created for each user consisting of observed sequences of commands executed consecutively on a single document. A series of commands as a "reading session" is defined for a particular user (identified by user ID). The commands were organized into sessions as follows. File names are not included in the data, thus individual files are identified using the file creation date and file size (in the dataset, the file size for a document remains constant as long as it is open). All commands belong to a single session as long as no more than 60 minutes elapse between issued commands.

The 74,783 commands coalesce to 11,334 sessions, an average of 6.59 commands per session. The longest session consisted of 93 commands, the shortest, one command. Three of the 24 authors participated in fewer than ten sessions. Data for these users are discarded. There are an average of 539 sessions per user. After organizing the data by reading sessions, all sessions belonging to an individual contribute to generate that user's command profile. The goal is to determine whether knowledge of a user's profile (the authenticated user), as well as knowledge from other users' profiles (masqueraders), is sufficiently distinctive to allow for differentiation between the authenticated user and masqueraders. This question is a natural fit for machine learning. To answer it, we compared the performance of commonly used machine learning algorithms with the following protocol:

```
For each user $u$:
    1) Label sessions belonging to $u$ as authenticated.
    2) Label sessions not belonging to $u$ as other.
    3) Perform a 10-fold cross validation (CV):
    4) For each fold $f$:
        Train classifier on all folds $\neq f$.
        Label each session in $f$ as authenticated or
        other.
    5) Evaluate average classifier performance across
       fields by calculating percent correct, F-measure
       and AUC.
```

In a 10-fold CV, the sessions are stratified into ten distinct folds consisting of 1/10th of the total data. Class distribution (the ratio of authenticated vs. unauthenticated sessions) is preserved in each fold. A classifier algorithm is trained on 9 of the 10 folds. The classifier is asked to label each session in the test fold as authenticated or other. This process is repeated 10 times, with each fold being used as test data once.

Each user is treated as the authenticated user once. Classifier performance is averaged across users. We evaluated several different well-known classifier algorithms including C4.5 decision tree, Naive Bayes, Adaptive Boosting (AdaBoost), and Random Forest. As a baseline we used a majority class classifier that always classified the instance as "other". The following metrics were used to compare classifier performance: percentage of correctly classified sessions, F-measure (combines precision and recall), and Area under the Curve (AUC) of the Receiver Operator Characteristic (ROC) curve (informs on hit rate vs. false accept rate). The larger those metrics are the better performance is.

As a follow up to this initial experiment, we have observed that the class distribution is extremely skewed. The number of sessions labeled "other" vastly outnumbers the number of sessions belonging to the authenticated class. We examined changes in classifier performance when other sessions are subsampled to 10% of their original number in the training data. The corresponding follow-up experiment also used a 10-fold CV for each user's profile as in the initial experiment.

V. Results

A. *Evaluating classifier performance*

We first compare the ability of the selected classifier algorithms to distinguish between the authenticated and other users (see Table II). The evaluation procedure by which each user profile is once treated as the authenticated user and then repeatedly as other is described in detail in the previous section. The Random Forest algorithm was trained with 20 decision trees trained on a randomly selected subset of eight features. AdaBoost was trained in 10 iterations with a decision stump as the base classifier.

TABLE II. AVERAGE ALGORITHM PERFORMANCE ACROSS USER PROFILES USING 10-FOLD CV. BASELINE PERFORMANCE FROM MAJORITY CLASS CLASSIFIER

Algorithm	Avg. percent correct	Avg. F-measure	Avg. AUC
Random Forest	95.43%	.943	.735
AdaBoost	95.07%	.931	.674
Naïve Bayes	95.01%	.932	.595
C4.5	95.42%	.939	.566
Baseline	95.04%	.927	.500

All classifiers outperformed the baseline, if only narrowly in some cases. Random Forest bested the other methods in terms of all three metrics. In particular, AUC was much higher for Random Forest than the other methods tested. Although the perceived increase in average percent correct is moderate for Random Forest, a comparison of a confusion matrix (see Table III and Table IV) between the second best algorithm AdaBoost and Random Forest reveals that Random Forest is identifying the minority/rare class (the authenticated user) at a better rate:

TABLE III. ADABOOST CONFUSION MATRIX

		Predicted class	
		Authenticated	*Other*
Actual class	*Authenticated*	3.49	52.79
	Other	3.29	1075.50

TABLE IV. RANDOM FOREST CONFUSION MATRIX

		Predicted class	
		Authenticated	*Other*
Actual class	*Authenticated*	10.59	45.66
	Other	6.19	1072.57

Random Forest and AdaBoost both perform well. Both are ensemble learning methods. Overall RF performs better due to its intrinsic characteristics, e.g., randomness and sampling (bagging), and ensemble methods, which cope better with the varying nature of individual user profiles. This level of optimal performance is in line with other research which has found that ensemble learning, either through boosting or bootstrap aggregating (bagging), are flexible enough to represent complex hypothesis functions that are difficult to learn with a single classifier. In addition, bagging (used by Random Forest) can reduce the concern that the learned model may over fit the dataset.

While these initial results suggest that it is possible to use application-specific commands to distinguish between authenticated users and others in Microsoft Word, the rate at which authenticated users are misclassified as other (a false negative) could be prohibitively high in practice (this would depend on the application and the action required when an active authentication system triggers a false negative). Subsection B examines a mechanism for reducing the false negative rate.

B. Subsampling the majority class

To reduce the false negative rate, the majority class was subsampled in the training set to 10% of its total size, bringing the number of sessions in the authenticated and other classes much closer to an even split. Results for this experiment are presented in Table V.

TABLE V. AVERAGE ALGORITHM PERFORMANCE ACROSS USER PROFILES WITH MAJORITY CLASS SUBSAMPLED TO 10%

Algorithm	Avg. percent correct	Avg. F-measure	Avg. AUC
Random Forest	85.76%	.879	.746
AdaBoost	87.56%	.876	.675
Naïve Bayes	58.54%	.614	.649
C4.5	85.12%	.871	.668

As one would expect, subsampling the majority class makes all the classifiers more sensitive to the minority class. Subsampling is only used in the training set, not the test set, as this would inaccurately represent the observed class distribution. Even though the average percent correct rate declines (by 9.67% for the Random Forest algorithm), average AUC actually increases. The relative significance of AUC comes from the derivation and interpretation of ROC where decision-making is a function of setting thresholds on similarity distances to trace the ROC curve.

Table VI presents the confusion matrix for the Random Forest algorithm trained on the subsampled training data. In comparison to the confusion matrix for the Random Forest algorithm (see Table IV) trained on the original training data, subsampling has made the classifier far more sensitive to classifying authenticated sessions as authenticated (true positives have increased). Subsampling the majority class thus results in a trade-off: a substantially lower false negative comes at the cost of potentially confusing additional other sessions as authenticated (an increase in false positives).

TABLE VI. RANDOM FOREST CONFUSION MATRIX WITH SUBSAMPLED TRAINING DATA

		Predicted class	
		Authenticated	*Other*
Actual class	*Authenticated*	36.96	17.01
	Other	144.06	936.47

VI. CONCLUSION

This paper proposes a new methodology for the problem of active or continuous authentication using command streams issued from GUI-based applications. We have demonstrated that like command line lexicon, GUI application command patterns can be used to create users' profiles that are identifiable. Past research has explored the idea of building users profiles based on their behavioral patterns when in interacting with such graphical interfaces especially in the area of human-computer interaction, it did so by analyzing the user's keystroke and/or mouse dynamics. However, none had explored the idea of creating profiles by capturing users' issued commands when interacting with a specific application, which goes beyond how a user strikes the keyboard or moves the mouse across the screen. It provides more dimensions to consider and a richer set of behavioral features to include when building behavioral profiles.

This paper utilizes a publicly available dataset [13] of user command streams generated from Microsoft (MS) Word usage to serve as a test bed. User profiles are first built using MS Word commands and discrimination takes place using machine learning algorithms. We report best performance using random forest (RF) and Adaboost with random forests coming first in terms of both accuracy and Area under the Curve (AUC) for the Receiver Operating Characteristic (ROC) curve. This was due, first, to implementing ensemble methods, with RF also characterized by a potent mix of randomness and subsampling vis-a-vis both the data samples chosen for training and the features chosen to represent the user profiles engaged in training. This is an essential adaptation strategy for active authentication to cope better with the varying nature of individual user profiles. The training strategy is further beefed-up using SMOTE to handle unbalanced populations, with imposters less prevalent.

Our future research will focus on exploiting the rich set of behavioral properties GUI-based application can reveal. Unlike CLIs, which limit the users' engagement to the set of commands they recognize, applications that are GUI-based allow for richer user interactions and could reveal a cognitive process that may help to infer a person's knowledge, intentions and, more importantly, identity. This should not be mistaken for what research in the area of human-computer interaction (HCI) have introduced thus far such as how a user moves his mouse, stroke keys or what processes are running. Our goal is to monitor application-specific properties of user's interaction. For example, consider how a user interacts with a document opened in MS Word. Besides the obvious possible commands, such as those used in our study (e.g., changing the font type, style and size), the way a user scrolls between pages, moves the cursor position or pauses at different sections of a document are valid and interesting set of actions. One can infer particular knowledge, context and intent (task) from such actions. We believe that such actions are observable and that discriminative behavioral and cognitive biometric signatures can augment user profiles.

Acknowledgment

Special thanks to Dr. Frank Linton, Principal Scientist at MITRE, for his interest in our research, and for making the dataset available to the public.

This research was supported by the Minister of Trade, Industry and Energy (MOTIE), Korea, under the Brain Scouting Program (HB609-12-3002) by the National IT Promotion Agency (NIPA).

References

[1] Aberdeen Group, "Strong User Authentication - Best-in-Class Performance at Assuring Identites," Mar. 2008.

[2] M. Karnan, M. Akila, and N. Krishnaraj, "Biometric personal authentication using keystroke dynamics: A review," *Appl. Soft Comput.*, vol. 11, no. 2, pp. 1565–1573, 2011.

[3] R. Katiyar, V. K. Pathak, and K. V. Arya, "A Study on Existing Gait Biometrics Approaches and Challenges.," *Int. J. Comput. Sci. Issues IJCSI*, vol. 10, no. 1, 2013.

[4] I. Traore, I. Woungang, Y. Nakkabi, M. S. Obaidat, A. A. E. Ahmed, and B. Khalilian, "Dynamic Sample Size Detection in Learning Command Line Sequence for Continuous Authentication," *Syst. Man Cybern. Part B Cybern. IEEE Trans. On*, vol. PP, no. 99, pp. 1 –14, 2012.

[5] D. R. KISKU, P. GUPTA, H. MEHROTRA, and J. K. SING, "Multimodal Belief Fusion for Face and Ear Biometrics," *Intell. Inf. Manag.*, vol. 1, no. 3, pp. 166–171, 2009.

[6] S. Noushath, M. Imran, K. Jetly, A. Rao, and G. Hemantha Kumar, "Multimodal biometric fusion of face and palmprint at various levels," in *Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on*, 2013, pp. 1793–1798.

[7] A. C. Morris, S. Jassim, H. Sellahewa, L. Allano, J. Ehlers, D. Wu, J. Koreman, S. Garcia-Salicetti, B. Ly-Van, and B. Dorizzi, "Multimodal person authentication on a smartphone under realistic conditions," in *Proc. SPIE conference on mobile multimedia/image processing for military and security applications, Orlando*, 2006.

[8] A. Azzini, S. Marrara, R. Sassi, and F. Scotti, "A fuzzy approach to multimodal biometric continuous authentication," *Fuzzy Optim. Decis. Mak.*, vol. 7, no. 3, pp. 243–256, 2008.

[9] T. Lane and C. E. Brodley, "Sequence matching and learning in anomaly detection for computer security," in *AAAI Workshop: AI Approaches to Fraud Detection and Risk Management*, 1997, pp. 43–49.

[10] J. Marin, D. Ragsdale, and J. Sirdu, "A hybrid approach to the profile creation and intrusion detection," in *DARPA Information Survivability Conference & Exposition II, 2001. DISCEX'01. Proceedings*, 2001, vol. 1, pp. 69–76.

[11] D. Y. Yeung and Y. Ding, "Host-based intrusion detection using dynamic and static behavioral models," *Pattern Recognit.*, vol. 36, no. 1, pp. 229–243, 2003.

[12] M. Schonlau, W. DuMouchel, W. H. Ju, A. F. Karr, M. Theus, and Y. Vardi, "Computer intrusion: Detecting masquerades," *Stat. Sci.*, pp. 58–74, 2001.

[13] F. Linton, D. Joy, H. P. Schaefer, and A. Charron, "Owl: A recommender system for organization-wide learning," *Educ. Technol. Soc.*, vol. 3, no. 1, pp. 62–76, 2000.

[14] "Microsoft Office Is Right at Home." [Online]. Available: http://www.microsoft.com/en-us/news/features/2009/jan09/01-08cesofficeqaschultz.aspx. [Accessed: 24-May-2012].

[15] P. Karcher, P. Burris, and T. Keitt, "Market Update: Office 2013 And Productivity Suite Alternatives," Forrester Research, Inc, Oct. 2013.

[16] "Active Authentication." [Online]. Available: http://www.darpa.mil/Our_Work/I2O/Programs/Active_Authentication.aspx. [Accessed: 18-May-2012].

[17] A. J. Klosterman, "Secure continuous biometric-enhanced authentication," DTIC Document, 2000.

[18] R. H. . Yap, T. Sim, G. X. . Kwang, and R. Ramnath, "Physical Access Protection using Continuous Authentication," in *2008 IEEE Conference on Technologies for Homeland Security*, 2008, pp. 510–512.

[19] D. Shanmugapriya and G. Padmavathi, "A survey of biometric keystroke dynamics: Approaches, security and challenges," *Arxiv Prepr. ArXiv09100817*, 2009.

[20] A. A. E. Ahmed and I. Traore, "A new biometric technology based on mouse dynamics," *Dependable Secure Comput. IEEE Trans. On*, vol. 4, no. 3, pp. 165–179, 2007.

[21] A. Weiss, A. Ramapanicker, P. Shah, S. Noble, and L. Immohr, “Mouse movements biometric identification: A feasibility study,” *Proc Stud. Res. Day CSIS Pace Univ. White Plains NY*, 2007.

[22] S. Hashiaa, C. Pollettb, M. Stampc, and M. Q. Hall, “On using mouse movements as a biometric,” 2005.

[23] K. Revett, H. Jahankhani, S. T. Magalhaes, and H. M. D. Santos, “A survey of user authentication based on mouse dynamics,” *Glob. E-Secur.*, pp. 210–219, 2008.

[24] A. A. E. Ahmed and I. Traore, “Detecting computer intrusions using behavioral biometrics,” in *Third Annual Conference on Privacy, Security and Trust, St. Andrews, New Brunswick, Canada*, 2005.

[25] A. Garg, S. Vidyaraman, S. Upadhyaya, and K. Kwiat, “USim: a user behavior simulation framework for training and testing IDSes in GUI based systems,” in *Simulation Symposium, 2006. 39th Annual*, 2006, p. 8–pp.

[26] R. A. Maxion and T. N. Townsend, “Masquerade detection using truncated command lines,” in *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*, 2002, pp. 219–228.