N-Gram-Based User Behavioral Model for Continuous User Authentication

Leslie Milton, Bryan Robbins, and Atif Memon,
Department of Computer Science,
University of Maryland, College Park, MD, USA
{lmilton,brobbins,atif}@cs.umd.edu

***Abstract*—We posit that each of us is unique in our use of computer systems. It is this uniqueness that we leverage in this paper to "continuously authenticate users" while they use web software. We build an n-gram model of each user's interactions with software. This probabilistic model essentially captures the sequences and sub-sequences of user actions, their orderings, and temporal relationships that make them unique. We therefore have a model of how each user typically behaves. We then continuously monitor each user during software operation; large deviations from "normal behavior" can indicate malicious behavior. We have implemented our approach in a system called *Intruder Detector* (ID) that models user actions as embodied in the web logs generated in response to the actions. Our experiments on a large fielded system with web logs of approximately 320 users show that (1) our model is indeed able to discriminate between different user types and (2) we are able to successfully identify deviations from normal behavior.**

Keywords–behavioral modeling; continuous authentication; software security; n-grams.

I. INTRODUCTION

The idea of continuous user authentication (CUA) is not new. The basic premise of CUA is that conventional user authentication, usually performed during the initial login session, is insufficient. In conventional authentication, users are not asked to verify their identity during their session, leaving the computer system vulnerable to malicious or unintended use while the user is logged-in. CUA techniques, on the contrary, monitor, verify, and authenticate users during their entire session. Functionally, CUA may be considered as one example of an intrusion detection system (IDS). CUA can be used to check the state of the system by continuously montioring user activity and comparing this activity with stored usage profiles to alert and/or de-authenticate the user when an intrusion is detected or suspected. IDS takes this one step further by adding a second line of defense to pinpoint misuse and initiate proper response [1]. Several studies have used biometrics to continuously authenticate users by the use of cognitive fingerprints, eye scans, color of user's clothing, and face tracking [2][3][4]. However, many of these techniques require additional hardware and cost to operate efficiently. Behavioral modeling addresses these limitations by monitoring how users interact with the system. Evaluating mouse movement, how a user searches for and selects information, and the habitual typing rhythm of users are traits used to continuously observe a user's behavior [5][6]. Although these approaches do not require special hardware, most of them do require the installation of specialized monitoring software.

In this paper, we posit we can obtain a unique behavioral footprint indicating patterns of use for a specific user or group of users of a particular web-based software using web log analysis. It is this footprint that we leverage to "continuously" authenticate the user. We can construct models of any targeted group of sessions based on n-grams. n-gram models have performed somewhat surprisingly well in the domain of language modeling, where researchers have found that a history of only one to two events is necessary to obtain optimal predictive capabilities [7]. An n-gram model captures all sequences and subsequences of a fixed length, N, from previously observed user input, which allows prediction and evaluation of future behavior based on frequencies. If we assume that the event sequences carried out by users of a software system are analogous to natural language, we would expect to find similar predictive power in n-gram models of software event sequences as well. To test the validity of our approach, we seek to answer the following research questions:

1) *Can we build discriminating user models to determine user types?*
2) *Can the model recognize various legitimate users who are operating in the same user session?*
3) *Can usage profiles be used to identify outliers in the user's behavior?*

Our approach is different from that employed by [3] because they focus on device usage and delays, mouse tracking, word usage, etc. Our approach also differs from various biometric approaches [8][4][9][10][11]. We instead focus on a particular layer of the software. With our approach, we evaluate web log data generated by users of a web-based system. The web logs are used to build unique profiles based on a user's role within the system. There is no need for additional hardware since we are using information that is automatically generated by the system. This process provides a continuous verification technique with no interaction from the user which not only improves security but enhances usability of the system.

We feel that our new approach should be used together with existing authentication mechanisms (e.g., passwords) to provide an increased level of security demanded by today's computing environment. ID is just one more tool in the security expert's toolbox.

This work makes the following contributions:

1) We use n-grams to model the behavior of users while they interact with web-based software.
2) We show how keywords are abstracted from web logs to develop user footprints.
3) We develop a continuous user authentication technique with the ability to categorize user sessions into a pre-defined set of roles or potentially finer-grained user profiles.

 ISBN: 978-1-61208-376-6

The rest of the paper is organized as follows: Section II provides details of related work. Section III defines the basis of continuous user authentication and how we model this activity using n-grams. Section IV describes our pilot study with experimental results. The conclusion and future work are discussed in Section V.

II. Background & Related Work

User authentication serves as a prevention-based method to protect malicious access of systems. However, if a malicious user is able to successfully pass the authentication step, there should be a transparent method to detect their behavior. For this reason, CUA is used as a second line of defense to check the state of the system by continuously monitoring user activity and simultaneously compares this activity with stored usage profiles to alert and/or de-authenticate the user. Finally, the IDS takes over and performs misuse and anomaly detection. The former identifies patterns of known attacks and the latter detects intrusions by determining whether there is some deviation from stored normal usage patterns.

Various research studies have explored the use of authentication. Kaminsky *et al*. address challenges for user authentication in a global file system [12]. This approach uses an authentication server to identify users based on local information. Researchers of cloud computing security methods have developed implicit authentication to identify a user's past behavior data to authenticate mobile devices [13]. These two studies have one major limitation worth noting. They lack the ability to continuously monitor user behavior for anomaly detection.

The realm of CUA has be extensively evaluated with the use of biometrics to verify user identity through the use of unique behavioral and/or phyical traits. A study by the Defense Advanced Research Projects Agency (DARPA) uses a combination of metrics that include eye scans and keystrokes to evaluate how the user searches and selects information [3]. In addition, a number of research studies concerning CUA use one or more hard and soft biometric traits to continuously authenticate a user. Niinuma *et al*. propose a CUA framework to automatically register the color of users' clothing and their face as soft biometric traits [4][2]. Results from this study show that the system is able to successfully authenticate the user with high tolerance to the user's posture. Limitations to these studies exist because of the additional hardware that is needed to implement this technique which can become costly if an entire organization uses this feature to authenticate users.

Altinok *et al*. propose a continuous biometric authentication system that provides an estimate of authentication certainty at any given time, even in the absence of any biometric data [10]. However, as the authentication uncertainty increases over time, system usability decreases. In a similar study, Kang *et al*. introduce temporal integration of biometrics and behavioral features to continuously authenticate users [14]. Similar to the previous biometric studies, additional hardware is needed.

A face tracking system that uses color and edge information has been used to compute behavioral features. Shen *et al*. use mouse dynamics when implementing continuous user authentication [5]. This technique is used to observe behavioral features in mouse operations to detect malicious users. However, there are some existing limitations with this emerging approach. Behavioral variability occurs because of human or environmental factors. Such changes could possibly identify the correct user as an impostor.

Our study extends beyond the aforementioned research studies in that: 1) Instead of using traditional biometric traits, we explore the possibility of using web log information that is automatically generated by web applications; 2) Our approach, integrated into a prototype tool, uses a novel and simple n-gram language model to capture user behavior; 3) Our experiments are based on data from users of a government system who are completing day-to-day tasks.

III. Continuous User Authentication

Our approach to CUA involves building n-gram models of user activity by observing sequences of user interaction with a web-based system. Once a model is constructed, we leverage it for the classification of incoming event sequences. However, the models are not without problems. While we have been able to address some challenges with tool support, other challenges remain. Below we formally define n-gram models, then present our approach to dealing with fundamental risks of using this type of model. Finally, we present algorithms for applying this model to the CUA domain.

A. n-gram Models

In general, n-gram models capture a probability distribution over some domain of events. As a simple example, imagine that we would like to track the probability of a Sunny (S) day or Rainy (R) day of weather. To learn the probability of a Sunny day, we could observe days for some period of time (*e.g.*, 10 days), and count the number of Sunny days observed, $nSunny$. Then, we could assign the likelihood of a Sunny day occurring to be $\frac{nSunny}{10}$. When waking up each morning, we assume that the probability of a Sunny day is given by this same fixed rate. Under this interpretation, to find $P(SSSS)$ (*i.e.*, five Sunny days in a row), we would simply solve $P(S)^5$. We can compute probabilities based on a set of given observations. The observations can be mapped to a series of class labels $\{w_0, w_1, w_2, ..., w_n\}$. Applying the chain rule of probability theory yields the probability of a sequence according to some prior context available at each data point:

$$\begin{aligned} P(w_1^n) &= P(w_1)P(w_2|w_1)...P(w_n|w_1w_2...w_{n-1}) \\ &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)...P(w_n|w_1^{n-1}) \\ &= \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \end{aligned} \quad (1)$$

The probability of $P(SSSS)$ would be given by

$$P(S) * P(S|S) * P(S|SS) * ...,$$

where $P(S|S_1..S_n)$ is the probability of S given we have seen the sequence $S_1..S_n$ immediately prior to S. When using this method, however, the number of parameters grow exponentially with the number of keywords in prior context. In some cases, we can reasonably apply the *Markov assumption*, which assumes that the probability of an event occurring is dependent only on the current "state" of a system. This state is defined as a fixed-length context or history, *h*.

n-gram models, then, are Markov models which use $(N-1)$ keywords of context to define the current state of the model.

 ISBN: 978-1-61208-376-6

Constructing, or training, an n-gram model requires the ability to observe example sequences occurring in the domain to be modeled. To train a model well, we need to observe single events from sequences in all relevant contexts.

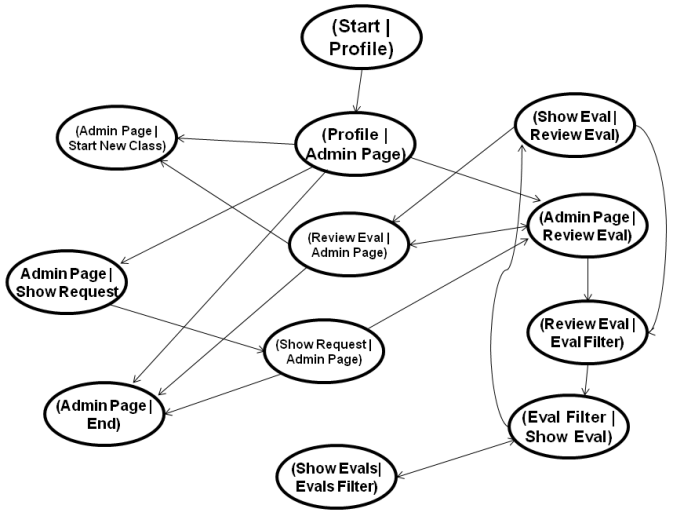

Figure 1. n-gram model of *User1* where n=3

In statistical analysis, it is sometimes difficult to estimate $P(w|h)$. Figure 1 represents an n-gram model of *User1*, where N=3 and *h*=2. In this figure, h is a sequence of the $(N-1)$ previous keywords. Probabilities are not included in this example. We deliberately keep the model simple by showing how current keywords depend on previous keywords when observing a sequence of actions. Since we are only concerned with a history of $h = 2$, the *(Start | Profile)* state is captured instead of *(Start)* as shown in Figure 1. As we transition from *(Start | Profile)*, we lose *Start*, but keep *Profile* to arrive at the *(Profile | Admin)* state. Now, every state that previously came from the *Admin* state, comes out of every *(<X> | Admin)* state. Once we construct an n-gram model in the training phase, we can use it for analysis of new event sequences in the test phase. With the *Markov assumption* incorporated into our construction of the model, we can apply the chain rule for conditional probabilities with a history of length $N-1$ to estimate the probability of any event sequence according to the existing model.

Finally, ID can be configured to incrementally improve its models during execution. Consider that User1 executes a new sequence of actions not contained in their n-gram model. This action can possible occur due to an update of the user interface, additiional duties added to User1, etc. ID will recognize the mismatch and ask for re-authentication (e.g., via a password, supervisoral approval). If this transaction is successful, then the sequence of actions that was not recognized is added to the n-gram model, thereby improving its accuracy. When fielded, we envision ID to be used in a training mode to build baseline models of all users; an then used in a deployment mode.

B. Challenges with n-gram Models

Even from a simple example, we see that additional data typically only stands to improve the reliability of a probabilistic model. With n-gram models, we capture conditional probabilities. The number of possible parameters needed by the model grows not only according to events observed, but also exponentially by the history being considered.

For the best possible model, our training phase requires observation of every possible event in every possible context. Additionally, we need to observe each context multiple times to have confidence that parameters are accurate. Without sufficient training, we may encounter event sequences during the test phase for which we have no knowledge to provide a probability estimate.

Others from the natural language domain have addressed this challenge of insufficient data for n-gram models. A concept known as *smoothing* involves saving some probability mass in an n-gram model's probability distribution for unseen events [15]. For the models in this paper, we have chosen to use the Kneser-Ney smoothing technique [16]. At a high level, Kneser-Ney involves:

- Reserving probability mass for unseen events by reducing all probabilities by a constant discounting percentage
- Estimate missing higher-order conditional probabilities by incorporating the observed frequencies of lower-order prefixes (a concept known as *backoff*)
- More precisely, combining the frequency of lower-order n-grams with a concept called a continuation probability, which estimates the probability that an event completes an n-gram, to estimate the probability of event sequences

For a more complete consideration of Kneser-Ney smoothing, the reader is referred to the original reference [16]. For our purpose, potential risks of applying Kneser-Ney to the domain of events carried out on software are violations of key features above. For example, if very few events are being performed by users, applying a constant discounting to every probability may save too much probability mass for unseen events. We will revisit the appropriateness of the model after gathering evidence in our experiments.

C. User Categorization

During the test phase of our experiments, we assign a probability to a sequence of events. We use binary categorization to judge a sequence as having likely been generated by a specific model (PASS) or not (FAIL). We introduce a probability threshold, t, for this pass/fail type of judgment for a sequence. Any sequence whose probability exceeds this threshold should be considered as a PASS, $+1$, and otherwise considered FAIL, -1.

A decison rule is used to predict the class membership of a given sequence of behavioral keywords, K. When new samples are encountered, the following decision rule is used:

$$\begin{cases} P(K,m) > t, & \text{then } y = +1 \\ P(K,m) < t, & \text{then } y = -1 \end{cases} \quad (2)$$

where $P(K,m)$ is the probability the behavioral keyword sequence is generated by the mth user's n-gram model. The probabilities are estimated using a training set of labeled data, $\{(m_0,y_0),(m_1,y_1),(m_2,y_2),...,(m_n,y_n)\}$, where label $y_i = \pm 1$ and depends on the class of m_i.

 ISBN: 978-1-61208-376-6

A more complex scheme that can also be useful for continuous authentication is multi-class categorization [17] [18]. For effective evaluation, this categorization method requires more training and test data. Under this decision-making approach, we can score an input sequence according to one of many models, and categorize the sequence as belonging to the model which estimates the highest probability. Therefore,

$$u = \arg\max_{m} P(K, m) \quad (3)$$

We use binary categorization by a simple threshold and multi-class categorization by comparing probabilities to translate n-gram models' estimations of sequence probability into decisions. We investigate our ability to make accurate decisions of various types as supported by these algorithms.

IV. Experiment

We now describe a set of experiments carried out on real user data designed to investigate the utility of n-gram models for CUA. In particular, we investigate the following research questions:

RQ1: Can we build discriminating user models to determine user types?

RQ2: Can the model recognize various legitimate users who are operating in the same user session?

RQ3: Can usage profiles be used to identify outliers in the user's behavior?

A. Subject System

We evaluate our proposed approach for CUA on an active government training support website for the high performance computing (HPC) community. This site is a repository for online training material which provides training course registration, course evaluation, information on several domain specific areas, and is a general source of information for its community of users.

Role	# of Users
Users	3775
Admins	6
Management	54
Technologist	2
Total # of users	**3837**

Figure 2. User Roles.

Each user account has an associated role. As shown in Figure 2, approximately 3800 users are in the "users" group which incorporates limited read access as well as the ability to evaluate and register for courses. These users do not access the system often. There are additional roles that provide access to areas of the system that are meant for administrative purposes; (*Admin, Management, Technologist*). The most prominent of these is the Admin role which has access to all areas. These users interact with the system often. Therefore, while we have more individual sessions available for the User role, the Admin role provides more per-user and per-session data. The Admin role also has the greatest risk for unauthorized use.

B. Data Collection

Because the system is web-based, access logs capturing hypertext transfer protocol (HTTP) requests made to the web server can provide sequences of user events. These logs are generated by the Tomcat JavaServer Pages (JSP) and servlet container. We analyzed the web log files for nine months to get accurate usage data for the system. User activity is largely based on role (i.e., access level). A user's role is monitored and used to verify their identity. This is under the assumption that users within the same role are likely to perform similar actions. System trust increases as the user interacts with the system if no outliers are identified in the CUA user model.

To validate data captured in a user's session, the following steps were used for preprocessing [19]:

1) *Data Cleaning*: The process of data cleaning is very important to generate an accurate picture of user activity when navigating a web application. For web logs of the subject system, various graphics and scripts are generated which add several entries to the log file. However, with our experiements, only JSP entries show user behavior and are important for logging purposes. Therefore, we remove all entries that are not related to user activity.
2) *User Identification*: Once the data is clean, the remaining file entries are grouped by individual user. Each user-id is associated with a role. User-ids are not continuously captured in the web log file. To solve this limitation, we check the session-id in a separate database table to capture unique user-ids.
3) *Session Identification*: Session identification is used to divide user accesses into individual sessions [19]. For the web access log, sessions are clearly identified. After checking the database for the role of each user, sessions are then grouped by role.
4) *Keyword Generation*: For each relevant JSP entry in the individual user log, a portion of the string is captured as a keyword. We are not considering parameters in this process because user entries would be too unique to categorize in a model.

When predicting individual user behavior (i.e., fine-grained approach), we filter the web logs to abstract only those users who have at least two sessions of activity and at least 80 keywords to ensure we have enough keyword data. After applying this filter, 31 users met this criteria and were used for this study. When predicting user types, 320 users were identified. In addition, at least two user roles must be present when predicting user types (i.e., user roles). To maintain the purity of the evaluation, we separate data into training and test sets, such that no model should be evaluated on its ability to classify data which was used during its own training.

C. Multi-class Categorization

For RQ1, we first want to consider whether unique profiles can indeed be constructed. To evaluate this research question, we develop models for each user role and use the multi-class categorization approach. Because we have four groups of users, a random model with no observation would achieve 25% accuracy in categorizing users. If our models are effectively capturing unique details about the behavior of users within roles, we would expect to see much greater overall accuracy

 ISBN: 978-1-61208-376-6

in categorization. We test the approach by comparing ID's recommended category to the actual category of each test session.

For the purpose of cross-validation, we performed various data splits for training and test data as seen in Figure 3 (50/50, 60/40, 70/30, 80/20, 90/10) to predict user roles. We reserve a precentage of sessions based on the data split to represent the testing set, E, and use the remaining data as the training set, R, to train an N order n-gram model for the specified class. We calculate $P(K, m)$ for each sample of keywords K from E. This represents the probability that the keywords are generated by the mth users n-gram model. Finally, m is selected with the highest probability for the behavorial keyword sequence, $P(K, m)$, and used as the prediction value. The y-axis, accuracy in Figure 3, shows this value for each data split.

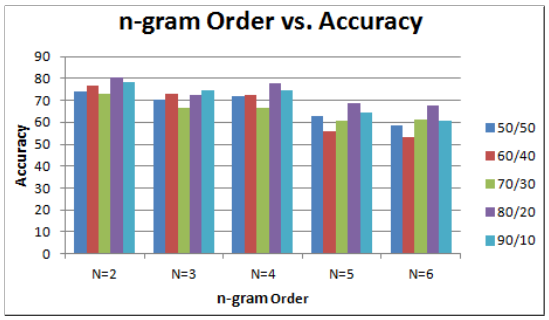

Figure 3. Prediction accuracy vs. n-gram order.

RQ1 results: Overall, the history order does not play a significant role. However, we do find that accuracy is highest when N=2 for each data split under evaluation when correctly categorizing test sequences into roles. Among each data split tested, the 80/20 data split has the highest accuracy, 80.5%, when N=2. The reported accuracies mark the mean performance over 10 separate trials, each of which chose random training data. For the 154 test sessions identified under this split, ID classified anywhere from 110 to 124 sessions correctly, depending on the sessions randomly chosen as training data. In most data splits, the accuracy began to decrease as the model order increased.

Based on these observations, we perform at a much greater rate of accuracy than a random selection of 25%, suggesting that at least some unique features of roles are effectively detected by the models. This shows using the models are both feasible and appropriate moving forward.

Instead of focusing on the four pre-defined user roles, RQ2 focuses on the ability of n-gram models to capture the behavior of specific users regardless of role. To evaluate this research question, we first filter the data under consideration to include only those users which have at least two sessions and at least 80 total keywords of input to ensure we have enough data to capture sequences. For the users meeting this criteria, we train models according to the n-gram approach.

RQ2 results: We achieved an overall accuracy rate of only 46% on the task of correctly categorizing test sequences by specific users. After filtering, we were left with only 28 test sessions. The tool correctly classified anywhere from 11 to 13 sessions correctly, depending on the sessions randomly chosen as training data. As in RQ1, the 46% overall mark represents the mean performance over 10 separate trials that were generated using random training data. We achieved our best results on this data when tuning model length to N=2 and implementing the 90/10 data split.

Working with so few test and training sessions, we have very little confidence in our evaluation of user specific profiles. In the future, we will be required to obtain much more data per user to effectively consider the construction of user-specific profiles.

D. Binary Categorization

When addressing RQ3, we want to consider whether the role-specific profiles constructed as part of RQ1 are capable of detecting outliers in user behavior. To evaluate this research question, we use the models from RQ1 independently in a binary categorization approach, as described in Section III-C. Because this task uses models independently, we use both training and test data from the remaining three roles to evaluate the model's ability to reject uncharacteristic sequences (i.e, negative examples). In addition, we use only test data from the model's own role to evaluate its ability to accept valid sequences (i.e., positive examples). Additionally, we would like to consider the effect of test sequence length (i.e., the number of keywords in an input sequence) on the performance of this task.

Finally, we track only a single threshold value for this task even though sequence length varies. A threshold which performs well for input sequences of length two would likely overestimate the threshold for longer lengths. As an alternative, we adapt the model's output probability to be an indicator of entropy, a measure of uncertainty in a system. This gives us the ability to normalize by the length of the input sequence. By doing so, we maintain a single threshold value for the binary classification task.

RQ3 results: Of all three research questions we consider, RQ3 relates most directly to the CUA goal of ID. Efficient experimental results are observed when accepting or rejecting snippets of user sessions based on role. We tested each model against every available subsequence of user keywords, from lengths two to nine with a probability threshold of -0.6. We obtained our best performance on this data when using N=9. Recall that backoff and smoothing in the model allow for the assignment of probabilities to a sequence of any length, regardless of the maximum history considered. Figures 4, 5, 6, and 7 show our findings. Note that accuracy is plotted separately for positive and negative test samples to analyze the models ability to reject uncharacteristic sequences and accept valid sequences.

We summarize the results as follows:

- As expected, the length-of-session examples provided to models significantly affects the ability to correctly classify the example.
- The effect of length was much greater on negative examples, as failures due to rejecting a positive sample were rare after lengths greater than four.
- The User model performed at a level greater than 90% on all samples for lengths greater than three.

 ISBN: 978-1-61208-376-6

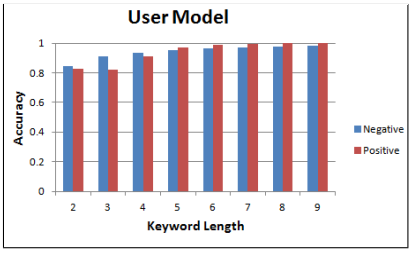

Figure 4. User n-gram model.

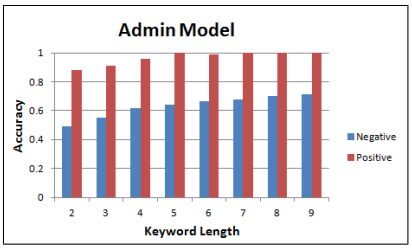

Figure 5. Admin n-gram model.

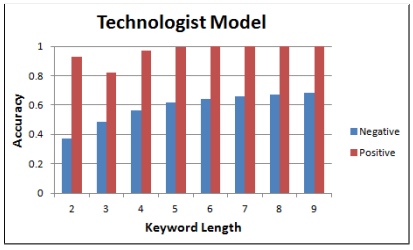

Figure 6. Technologist n-gram model.

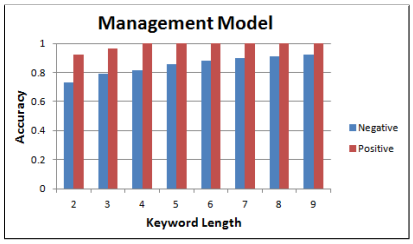

Figure 7. Management n-gram model.

- The Management model eventually achieved performance of 92%, though this took sessions of length nine.
- The Admin and Management models averaged 71% and 68% on negative examples, respectively, even at the maximum considered length of nine.

From these results, we can conclude that binary categorization proves to be much more effective than a random baseline at detecting uncharacteristic user behavior. For two of the four models considered, we achieved above 90% correct identification of negative samples and 100% correct acceptance of positive samples. In particular, the finding that User sessions can easily be protected against uncharacteristic usage is promising. Due to a large data set and elevated level of privileges for tasks, we expected the Admin user role to have one of the strongest models but this was not observed. In general, n-gram models seem much better suited for binary categorization tasks such as this one, especially given limited amounts of available data. In the future, perhaps multi-class categorization problems could be restated as a series of binary decisions.

Alternatively, the improvement in performance could be due to the use of shorter sessions which are less likely to contain unseen events. In this case, we rely on the validity of smoothing assumptions for accurate probability estimation. In the future, we will consider the effect of test example length on other tasks performed by ID as well.

V. Conclusions & Future Work

In this paper, we propose a continuous probabilistic authentication approach to model the behavior of users that interact with web-based software. A significant advantage of this mode of authentication is that it can be employed throughout the period of interaction, and hence, provide a natural way to continuously authenticate users. We have built a prototype, *Intruder Detector*, which keeps track of user actions, builds user profiles based on the n-gram language model and use these models to discriminate between user roles, and potentially finer-grained user profiles. Results show that *Intruder Detector* achieves 80.5% accuracy in user role classification tasks and nearly perfect accuracy when identifying categorization errors.

Although our pilot study has shown promising results, much work remains. In the immediate short term, we intend to work with a larger data set and compare various smoothing techniques. This will help improved the accuracy of RQ2 to correctly categorizing individual users. Capturing more data will help to better understand the characteristics of good, well-trained user models. We also plan to work out the details of fielding ID and evaluate two modes of operation: *training* in which models get built; and *deployment* in which the models get used for CUA. Afterwards, we will evaluate an alternative fielding strategy, one in which we have another level of authentication, using conventional means, in case ID identifies an intruder; this strategy will allow the models to iteratively get better (i.e., more accurate) with time.

Acknowledgment

This material is based on research sponsored by DARPA under agreement number FA8750-14-2-0039. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

[1] J. Liu, F. Yu, C.-H. Lung, and H. Tang, "Optimal combined intrusion detection and biometric-based continuous authentication in high security mobile ad hoc networks," Wireless Communications, IEEE Transactions on, vol. 8, no. 2, Feb 2009, pp. 806–815.

[2] K. Niinuma, U. Park, and A. K. Jain, "Soft biometric traits for continuous user authentication," Trans. Info. For. Sec., vol. 5, no. 4, Dec. 2010, pp. 771–780. [Online]. Available: http://dx.doi.org/10.1109/TIFS.2010.2075927

[3] R. P. Guidorizzi, "Security: Active authentication," IT Professional, vol. 15, no. 4, 2013, pp. 4–7.

[4] K. Niinuma, A. K. Jain, J. B. Kumar, S. Prabhakar, and A. A. Ross, "Continuous user authentication using temporal information," SPIE., vol. 7667, 2010.

 ISBN: 978-1-61208-376-6

[5] C. Shen, Z. Cai, and X. Guan, "Continuous authentication for mouse dynamics: A pattern-growth approach," in Proceedings of the 2012 42Nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), ser. DSN '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 1–12. [Online]. Available: http://dl.acm.org/citation.cfm?id=2354410.2355184

[6] F. Monrose and A. D. Rubin, "Keystroke dynamics as a biometric for authentication," Future Gener. Comput. Syst., vol. 16, no. 4, Feb. 2000, pp. 351–359. [Online]. Available: http://dx.doi.org/10.1016/S0167-739X(99)00059-X

[7] D. Jurafsky and J. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 2nd ed. Pearson Prentice Hall, 2009.

[8] S. Zhang, R. Janakiraman, T. Sim, and S. Kumar, "Continuous verification using multimodal biometrics," in Proceedings of the 2006 International Conference on Advances in Biometrics, ser. ICB'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 562–570. [Online]. Available: http://dx.doi.org/10.1007/11608288_75

[9] A. Azzini and S. Marrara, "Impostor users discovery using a multimodal biometric continuous authentication fuzzy system," in Proceedings of the 12th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Part II, ser. KES '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 371–378. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-85565-1_47

[10] A. Altinok and M. Turk, "Temporal integration for continuous multimodal biometrics," in In Multimodal User Authentication, 03, pp. 11–12.

[11] A. J. Klosterman and G. R. Ganger, "Secure continuous biometric-enhanced authentication," Tech. Rep., 2000.

[12] M. Kaminsky, G. Savvides, D. Mazieres, and M. F. Kaashoek, "Decentralized user authentication in a global file system," in Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, ser. SOSP '03. New York, NY, USA: ACM, 2003, pp. 60–73. [Online]. Available: http://doi.acm.org/10.1145/945445.945452

[13] R. Chow, M. Jakobsson et al., "Authentication in the clouds: A framework and its application to mobile users," in Proceedings of the 2010 ACM Workshop on Cloud Computing Security Workshop, ser. CCSW '10. New York, NY, USA: ACM, 2010, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1145/1866835.1866837

[14] H. B. Kang and M. H. Ju, "Multi-modal feature integration for secure authentication," in Proceedings of the 2006 International Conference on Intelligent Computing - Volume Part I, ser. ICIC'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 1191–1200. [Online]. Available: http://dx.doi.org/10.1007/11816157_148

[15] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in Proceedings of the 34th annual meeting on Association for Computational Linguistics, 1996, pp. 310–318.

[16] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on, vol. 1, 1995, pp. 181–184 vol.1.

[17] Y. Liu, Z. You, and L. Cao, "A novel and quick svm-based multi-class classifier," Pattern Recogn., vol. 39, no. 11, Nov. 2006, pp. 2258–2264. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2006.05.034

[18] P. Honeine, Z. Noumir, and C. Richard, "Multiclass classification machines with the complexity of a single binary classifier," Signal Processing, vol. 93, no. 5, 2013, pp. 1013 – 1026. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165168412004045

[19] R. Cooley, B. Mobasher, and J. Srivastava, "Data preparation for mining world wide web browsing patterns," KNOWLEDGE AND INFORMATION SYSTEMS, vol. 1, 1999, pp. 5–32.

 ISBN: 978-1-61208-376-6