2015 IEEE International Conference on Software Quality, Reliability and Security

An Effective Approach to Continuous User Authentication for Touch Screen Smart Devices

Arun Balaji Buduru and Stephen S. Yau
Information Assurance Center, and
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe, AZ, USA
{abuduru, yau}@asu.edu

Abstract: **Due to the rapid increase in the use of personal smart devices, more sensitive data is stored and viewed on these smart devices. This trend makes it easier for attackers to access confidential data by physically compromising (including stealing) these smart devices. Currently, most personal smart devices employ one of the one-time user authentication schemes, such as four-to-six digits, fingerprint or pattern-based schemes. These authentication schemes are often not good enough for securing personal smart devices because the attackers can easily extract all the confidential data from the smart device by breaking such schemes, or by keeping the authenticated session open on a physically compromised smart device. In addition, existing re-authentication or continuous authentication techniques for protecting personal smart devices use centralized architecture and require servers at a centralized location to train and update the learning model used for continuous authentication, which impose additional communication overhead. In this paper, an approach is presented to generating and updating the authentication model on the user's smart device with user's gestures, instead of a centralized server. There are two major advantages in this approach. One is that this approach continuously learns and authenticates finger gestures of the user in the background without requiring the user to provide specific gesture inputs. The other major advantage is to have better authentication accuracy by treating uninterrupted user finger gestures over a short time interval as a single gesture for continuous user authentication.**

Keywords: Touch-screen smart devices, adaptive continuous user authentication, reinforcement learning, and user re-authentication.

I. INTRODUCTION

Smart devices are experiencing explosive growth in terms of the number in use as well as the services offered by vendors. According to a recent report [1], vendors shipped about two billion touch-screen smart devices in 2013, including smartphones and tablets, and it is projected that the number would exceed 2.8 billion by the end of 2016. The information available in personal smart devices, such as smartphones and tablets, includes more detailed and precise information on their users' personal attributes than the information in other computing devices since personal smart devices are rarely shared among different users. Proliferation of applications across various domains, such as banking, shopping, and healthcare, requires smart devices to store even more sensitive user data. Furthermore, the rising BYOD (Bring Your Own Device) trend is leading to storage of enterprise data on personal smart devices [2]. The increasing number of sensors in smart devices make them sources of rich personal data of the users, like frequently visited locations. All these trends make personal smart devices more attractive targets for malicious activities, ranging from physically stolen devices to infected smart devices by malware.

Smart devices are traditionally protected by authentication methods using PIN, passphrase, pattern, face recognition and fingerprint. One of the major weaknesses of these authentication methods is that they do not continuously authenticate users after the initial authentication [3]. Hence, if a smart device is physically compromised (including being stolen) after the authentication has been performed, the attacker can keep the authenticated session open and extract the confidential data before the session times out or the user shuts down the device. Smart devices are usually small and make them particularly vulnerable for theft. Although there are various ways to remotely locate, erase and lock a smart device in case of theft, all of these mechanisms can be circumvented by methods, like removing the SIM card and disconnecting access to internet. Smartphones pose additional challenges to t-size he existing authentication mechanisms, such as having small screen size and computational resources. Small screen size prevents users from using "strong" passwords because it normally requires a mixture of letters, numbers and symbols which are not present in existing smart device lock screen.

Hence, a mechanism, that authenticates the user continuously through the entire usage session which is unobservable to the user, is needed. In this paper, we will present an approach using Markov decision process (MDP) and touch gestures of the user to continuously authenticate the user on a touch-screen smart device. This approach will include algorithms to estimate the degree of importance of each cell on the touch screen of the user's smart device for continuous authentication, and to identify the usage context of the smart device. This approach has two major advantages. One is that this approach is more efficient by adaptively updating the authentication model with evolving user's finger gestures. The other major advantage is to have better authentication accuracy by treating uninterrupted user finger gestures over a short time interval as a single gesture for continuous user authentication.

The organization of this paper is as follows: After discussing the current state of the art in Section II, the features used for continuously authenticating a smart device's user will be presented in Section III. A method of estimating the degree of importance of each cell on the touch screen of the user's smart device to continuous user authentication will be presented in Section IV. The estimation of usage context of the smart device based on user's touch gestures will be presented in Section V. In Section VI, we will present a technique using the MDP for continuous user authentication. Our approach and its process flow will be summarized in Section VII. An example to illustrate a part of our approach will be given in Section VIII, and conclusions and future research will be discussed in Section IX.

II. CURRENT STATE OF ART

Most of the current smart devices use the authentication factors to authenticate the legitimate user of a smart device, and these authentication factors have been generally classified into three categories: *what you know*, *what you have* and *what you are* [3]. The authentication factors belonging to the categories of *what you know* and *what you have* are not suitable for continuous user authentication of smart devices because the user needs to frequently provide conscious inputs for continuous authentication, which will likely cause undesirable user experience. This leaves us with only the category of *what you are*. Even in this category, some factors, such as fingerprints, iris and face recognition, are inherently infeasible due to the requirements of additional hardware in low-cost touch-screen smart devices, and frequent conscious user interactions. Furthermore, existing approaches [5-9] to continuous user authentication use centralized architecture, where a server collects specific user data for a specific time interval in the training phase, generates the authentication model from the user data, and uses an authentication model to authenticate the user during subsequent usage. If the user behavior changes, then the user needs to undergo the training phase again so that the authentication model for the user is updated with the latest information of the user's behavior. Hence, this process has inherent security, privacy, overhead and scalability problems

Password authentication schemes fall into the *what you know* category, and the hardware dongles, like RSA SecurId, fall into the *what you have* category. Biometric authentication schemes, such as fingerprint, iris and face-pattern recognition, fall into the *what you are* category. The recent trend of authentication systems involves two-factor authentications, in which the password is usually accompanied by a one-time sign in code sent to the smart device. This trend combines the two authentication factors, *what you know* (password) and *what you have* (mobile device), to provide a slightly better authentication. The inherent problem with this two-factor authentication system is that in each of these two factors, sensitive information is lost in the event of theft of the smart device or attacker(s) breaking the two-factor authentication scheme.

Continuous user authentication schemes using the authentication factor *"What you are"* can be broadly categorized into two groups: using touch gestures [5-6] or multi-modality classification focusing on the data from a variety of sensors to model the user behavior [7-9]. Most of the continuous authentication techniques in both categories involve an enrollment (training) phase and an authentication phase. The system learns the necessary patterns from the user's related biological and/or sensory data, and stores them in the enrollment phase. In the authentication phase, the system compares the observed user related biological and/or sensory data against the patterns stored during the enrollment phase to re-authenticate a user.

Niu and Hao [4] conducted continuous user authentication experiments on iPad using user behavioral features in multi-touch operations, involving 34 volunteers. Their experiments showed Equal Error Rates (EERs) of 7%-15% for one mode of multi-touch and EERs of 2.6%-3.9% if two multi-touch modes are combined. Frank, et al, [5] presented an approach to continuous user authentication based on their touch gestures with EERs in the range up to 4% using a k-Nearest Neighbors (k-NN) classifier and a Support Vector Machine (SVM) classifier. They demonstrated this approach to extract 30 features from a dataset of 41 users based on users' swipe/scroll gestures. Later, Li, et al, [6] presented a touch-based authentication technique on Motorola Droid phones running Android 2.2. This approach leverages the device files being used by Linux Multi touch protocol to monitor all the touch gestures of the user on the smart phone. This method was evaluated using SVM on 75 users who were allowed to use the smart phones freely, and showed the average classification accuracies of 95%.

For the approaches combining multiple biometric inputs to produce aggregated user identification results, Muncaster and Turk [7] presented an approach to performing continuous, score-level multi-modal authentication based on a weighted sum of scores from each modality. A continuous multi-modal biometrics system using a hidden Markov model (HMM) was developed by Sim, et al, [8]. It includes integration of the results from a fingerprint biometric classifier with a face classifier to improve the accuracy of continuous user authentication. Shi, at el, [9] used multi-modal inputs, such as voice, location, multi-touch and motion, to perform continuous user authentication. Using Naïve Bayes classifiers on the features of multi-modal inputs, they showed average accuracy of over 95%.

All the above techniques are based on the assumption of zero-effort threats, where the adversary is assumed to be incapable of pulling off advanced forgery attacks [10] on the system, which are based on the characteristics of the user behavioral biometric patterns exhibiting large intra-user variation and overlap across a large population. The attackers leverage these characteristics to extract users' statistics from a large population database and perform advanced forgery attacks which can defeat these authentication techniques. The attack technique using robotic arm [10] increases the EERs in the continuous authentication techniques. In order to thwart these kinds of attacks, it is required to capture more unique characteristics of the user's finger gestures for differentiating the legitimate user from others. This increases the effort required on the part of the attackers to learn the user's behavior from a large population database, which may act as an effective deterrence.

Based on the above discussion, the existing continuous user authentication techniques fail to capture the nuances of the user's gestures, and they can only capture the user's gestures during the training phase. This will lead the continuous authentication techniques to miss some important strains of the user's behavior exhibited outside the training phase. This is particularly dangerous because, if the user model does not capture the unique characteristics of the user gestures, an attacker with access to a large set of normal user data might be able to construct a model conforming to the characteristics of the majority of the features exhibited by the population. This will enable the attacker to break the continuous authentication techniques. In order to capture a majority of unique characteristics of the user's gestures on a smart device, the technique needs to incorporate continuous learning of the user's gestures. In our approach, we will use a continuous authentication technique, which can continuously update the user's finger gesture model to effectively authenticate the legitimate user.

III. REPRESENTATION OF USER FINGER GESTURES ON SMART DEVICE TOUCH SCREEN

In our approach, we use two kinds of user gestures; one is micro-gesture which is uninterrupted user's finger movement on the smart device screen, and the other is the macro-gesture which is a collection of micro-gestures over a preset time interval. We will use the data acquisition procedure [6] for extracting raw user's finger gesture data from the touch screen of the smart device.

In order to efficiently represent the user's finger gestures on the touch screen of the smart device, the set of features needs to be simple, and yet can represent the various finger gestures reflecting the user's unique usage characteristics of the smart device. This will ensure that the collected information from the touch screen of the smart device is sufficient to distinguish the legitimate user of a smart device from other users, including malicious users. In our approach, we select dynamic and unique parameters, such as user's finger touch pressure and speed of user's finger gestures. We set two levels of features, called primarily and secondary, from the features used in the smart device authentication techniques [6] to ensure the efficient representation of user's finger gestures and accurate continuous authentication of the legitimate user. Our illustration in Section VIII shows that these features are reasonably good and sufficient for continuous user authentication.

1) *Primary features:*
   a) *First touch (FT) of a cell,* represents the cumulative total number of times the user's micro-gesture started on the cell of the smart device.
   b) *Duration of touch (DT) of a cell,* represents the duration of all the user's touch micro-gestures on a particular cell of a smart device touch screen.
   c) *Pressure of touch (PT) of a cell,* represents the pressure of all the user's touch micro-gestures on a particular cell of a smart device touch screen
   d) *Frequency of usage (FU) of a cell,* represents the cumulative total of particular cell accessed in a user's macro-gesture on the smart device touch screen.
2) *Secondary features:*
   a) *Average touch pressure (APT) of a cell,* represents the cumulative average touch pressure on a particular cell in a user's macro-gesture on the smart device touch screen. This feature is derived from the *PT*.
   b) *Average Duration of Touch (ADT) of a cell,* represents the cumulative average touch duration on a particular cell in a user's macro-gesture on the smart device touch screen. This feature is derived from the *DT*.
   c) *Speed of user gestures (SG),* represents the average speed of the user's micro-gestures in a particular macro-gesture on the smart device touch screen. This feature is derived from the *DT*.

IV. ESTIMATION OF IMPORTANCE OF CELLS OF SMART DEVICE TOUCH SCREEN FOR CONTINUOUS USER AUTHENTICATION

In this section, we will present the algorithm used in our approach to generate the reward value R_{C_i} of each cell of touch screen of the smart device. The R_{C_i} represents the importance of cell and is estimated using the primary and secondary features of the touch screen of a smart device. Each cell of touch screen of the smart device is a state in the state graph of MDP [11, 12]. The MDP can be represented by a 4-tuple:

$$< S, A, T, R >,$$

where S represents a finite set of cells on the user's smart device touch screen, A represents a finite set of actions, which represent the user's finger gestures, possible from every cell on touch screen of the smart device. T represents the set of probabilities $PR_{C_i}^{E_j}$ for the occurrence of every action E_j from each cell C_i. R represents the reward value R_{C_i} of C_i, which is derived from the features of the legitimate user's finger gestures on the smart device's touch screen as described in (5). The R_{C_i} is used as a metric for estimating the degree of importance of each cell on the user's smart device's touch screen.

The collection of all the cells of a smart device's touch screen in a time period T_x, is denoted by S_{T_x}, can be represented as follows:

$$S_{T_x} = < C_1, C_2, \dots, C_i > \qquad (1)$$

where C_i is a cell of the touch screen of the smart device, where i = 1, 2, …, n, and *n* is the number of cells on the smart device's touch screen, and C_i is a 4-tuple

$$C_i = < FT_i, ADT_i, APT_i, FU_i > \qquad (2)$$

where $FT_i, ADT_i, APT_i, and\ FU_i$ represent the features first touch, average touch duration, average touch pressure, and frequency of usage of C_i respectively.

$$D\left(C_i, CX_{T_x}\right) = \frac{FT_i \times ADT_i \times APT_i \times FU_i}{CX_{T_x}} \qquad (3)$$

$$S\left(e, T_x, CX_{T_x}\right) = \frac{e \times CX_{T_x}}{T_x} \qquad (4)$$

The $D\left(C_i, CX_{T_x}\right)$ and $S\left(e, T_x, CX_{T_x}\right)$, given in (3) and (4), are used for measuring the importance of the cell and the importance of usage context to continuous user authentication, respectively. The CX_{T_x} will be defined and calculated using (6) in Section V. The larger value of $D\left(C_i, CX_{T_x}\right)$ indicates that the specific C_i is more important in terms of user's touch pattern. The larger $S\left(e, T_x, CX_{T_x}\right)$ indicates that the user's finger gestures can provide more robust continuous authentication. $D\left(C_i, CX_{T_x}\right)$ and $S\left(e, T_x, CX_{T_x}\right)$ are then used for calculating R_{C_i}.

The R_{C_i} of the C_i of the smart device's screen are calculated as follows:

$$R_{C_i} = aD(C_i, CX_{T_x}) + bS(e, T_x, CX_{T_x}) \qquad (5)$$

where a, b and e are the weighting factors initialized during the application installation based on the type, the screen size, location, usage context, computational and space constraints of the smart device.

(5) is derived from our objective trade-off function (OTF) [13] for quantitatively measuring and incorporating performance and security in service-based systems. Since security configuration and traffic frequency vectors used in OTF are very similar to our primary and secondary features, we adapt the OTF to estimate the values of the cells based on the user's performance and security metrics requirement.

$PR_{C_i}^{E_j}$ is the probability metric for every edge E_j of C_i and is generated based on the user finger gesture movements on the smart device's touch screen. All the secondary features will be used for generating the update frequency of the probability and reward metrics in the state graph of the MDP because the secondary features are more stable and more accurately represent user's macro figure gestures. A major advantage of our approach is that it is flexible and designed to work even if a different technique is used to generate the R_{C_i} and $PR_{C_i}^{E_j}$ depending on the user's application requirements.

V. IDENTIFICATION OF USAGE CONTEXT OF USER'S SMART DEVICE

In this section, we will present our technique to identify the usage contexts, which indicate how the user, in terms of finger gestures, is using the smart device over the time period, such as gaming, reading, and streaming videos. The usage context of the smart device is very important because it affects the nature of the user gestures. In order to have good accuracy for user authentication, it is essential to identify the smart device's usage context. On the other hand, the existence of different usage contexts implies that we need to change the MDP models to facilitate continuous authentication for the different usage contexts. Hence, it is desirable to have a small number of usage contexts for the user's smart device for quickly authenticating the legitimate user. From authentication point of view a larger number of usage contexts may not necessarily improve user authentication accuracy.

In our approach, we classify the user gestures into n usage contexts, where n is a preset number of usage contexts for a specific application. The classification will be done based on the primary and the secondary feature inputs. The idea here is to basically identify the usage context of the smart device usage. Each usage context has unique characteristics such as in gaming mode the user may use the touch screen of a smart device more frequently compared to other usage contexts. We try to use the primary and secondary features to discriminate the user gestures based on their unique characteristics and classify them. The equation for identifying the context is shown in (6),

$$CX_{T_x} = eNS(APT_i, ADT_i, SG_i) \qquad (6)$$

where CX_{T_x} is the usage context of the smart device's touch screen for a time period T_x, and APT_i, ADT_i, SG_i are the secondary features as mentioned in Section III. The function NS is a normalized summation function which adds the normalized values of $APT_i, ADT_i\ and\ SG_i$. The major advantage of our approach is that it is flexible and designed to work even if a different technique is used to generate the CX_{T_x} based on the application requirement. The value of CX_{T_x} is then used to identify the smart device's usage context.

VI. GENERATION OF PLANS FOR CONTINUOUS USER AUTHENTICATION

In this section, we will discuss how to generate the plans [14] used for continuously authenticating the user of the smart device. Each plan is a sequence of states in a MDP state graph and represent a particular user's micro-gesture on the touch screen of the smart device. The continuous authentication technique for authenticating the legitimate user continuously, is performed using the generated plans for all the micro-gestures. The cells on the touch screen of the smart device are used to capture the user's finger gestures. The MDP tuples described in Section IV are specifically modeled to meet the requirements for our continuous authentication approach. The application provider initially sets the probability and reward metric of every edge and cell in the state graph respectively, based on the typical user finger gesture information. These parameters are then initialized on user's smart device during application installation or new user registration procedure. Based on the user's smart device usage, the parameters will be personalized to the individual user using the equations discussed in Section IV. Each user's usage context of the smart device has its respective probability and reward metrics.

In our approach, we use policy iteration technique [15] and value iteration technique [16], each of which is used to generate the plans for all the micro-gestures, because they are more efficient, accurate and relevant to perform continuous user authentication in a smart device. We have compared the performance of these two techniques using the illustrated example in Section VIII. Other popular MDP techniques, such as modified policy iteration technique [17] are not suitable as they need more accurate and stable heuristics to use effectively, and generating these accurate and stable heuristics is very difficult because of the dynamic nature of the user finger gestures.

The value iteration technique [16] enables the MDP to constantly perform backward induction using the Bellman backup equation to generate the plans. In the value iteration technique, there is no initial plan and the plan is generated and then updated when the reward values of the cells are updated. The reward values for all the cells are initialized using (5) based on the degree of importance the typical user finger gestures, and is then adjusted continuously based on the legitimate user's finger gestures. The value iteration technique will generate plans that maximize the cumulative reward values of the plans in a probabilistic environment. The cumulative reward value of a plan is the summation of the reward value of all cells in a plan. The value iteration technique, after generating the plan, will keep updating the reward values of the cells till the plans stabilize. This is done to dynamically update the degree of importance of all the cells based on their neighboring cell reward values. This enables the system to be more accommodating to legitimate user's errors, thus reducing the false negative rates.

The policy iteration technique [15] works similarly to value iteration technique, but the difference is mainly that in policy iteration technique we initialize the technique with pre-selected plans and then use the backward induction to update the plan. The output of the policy iteration technique is a set of plans along with their respective cumulative reward values. The major advantage of the policy iteration technique with respect to the user continuous authentication is that unlike the value iteration technique, the policy iteration technique has prior knowledge on the plans, and hence converges quickly. Policy iteration technique is more likely to achieve plan convergence faster than the value iteration because the policy iteration technique iterates over the plans instead of the individual reward values of state as in the case of the value iteration technique. But, if the initial plan assignment is random or incorrect, then the policy iteration technique may take longer to converge or generate suboptimal plans, either of which will affect the continuous user authentication.

In summary, each of the value iteration technique and the policy iteration technique will generate a set of plans along with their cumulative reward values. These plans will then be ranked based on their cumulative reward value, importance to the observed user's micro-gestures and the smart device usage context. These plans can be pre-computed on the user's smart device when it is not being used actively. The cumulative reward value of the ranked plans are then used as benchmark data for continuously authenticating the user's macro-gestures on the touch screen of the smart device. The dynamic nature of the MDP enables the reward metrics and the probabilities to keep changing continuously along with the user's evolving gestures. This is unlike the supervised learning techniques, where the learned pattern does not change with the evolving user finger gesture.

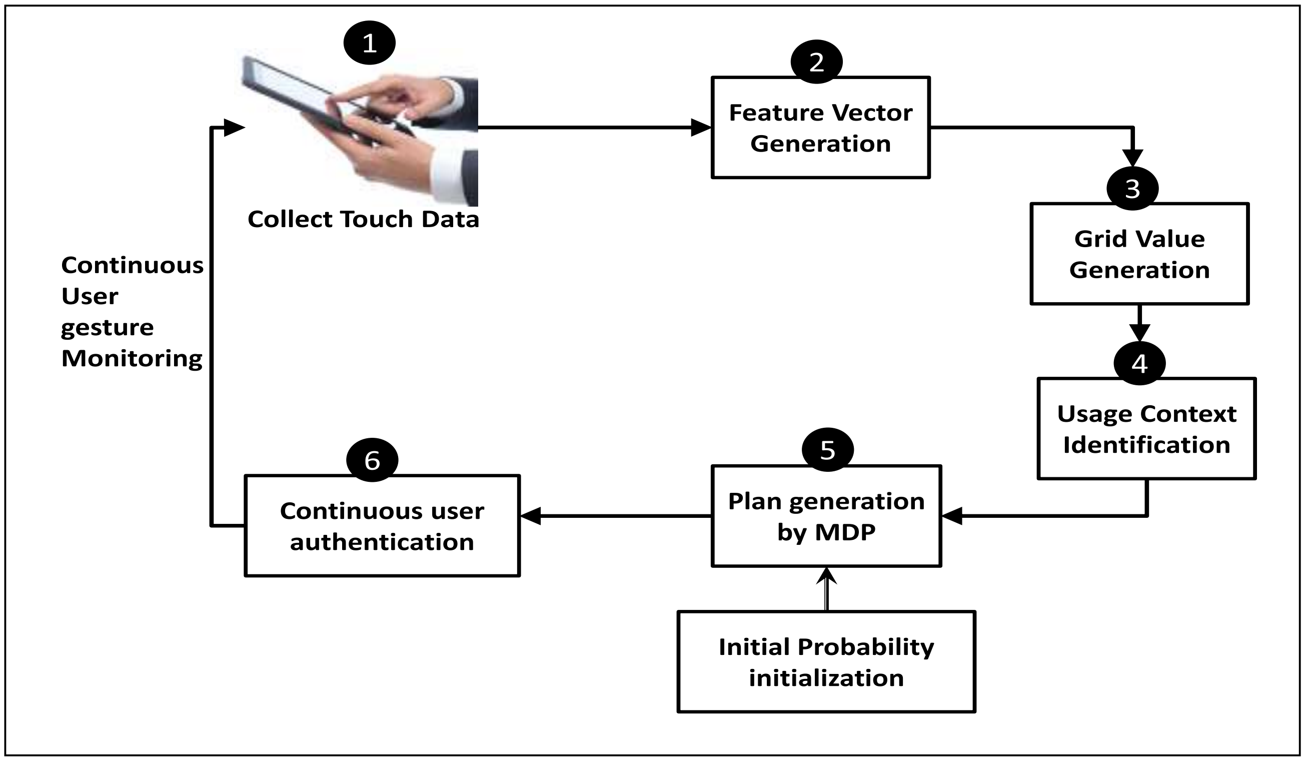
1
Collect Touch Data
2
Feature Vector Generation
3
Grid Value Generation
4
Usage Context Identification
5
Plan generation by MDP
Initial Probability initialization
6
Continuous user authentication
Continuous User gesture Monitoring

Figure 1: Process flow of our approach to continuous user authentication for touch-screen smart devices

The applicability of each of these two techniques depends on the user's application requirements. In some cases, one technique is better than the other and vice versa. Since the policy iteration technique has the prior typical user finger gesture information, the technique can perform continuous user authentication faster, but with relatively acceptable error rates which may be suitable for normal user authentication. However in highly critical applications, more robust continuous user authentication is needed. In such applications, the value iteration technique will be more suitable because it has no biases on user finger gestures and hence can achieve more accurate continuous user authentication.

VII. OUR APPROACH

In our approach we assume that the attacker is very sophisticated and has the following capabilities:

- Able to observe the behavior of the user on the smart device (shoulder surfing)
- Able to physically steal user's smart device
- Able to derive user's smart device authentication pin

In addition, we assume that all the states of a smart device are known and observable because in our approach the observable smart device touch-screen is divided into cells of a grid and each cell represents a state in the MDP state graph. Since all the required cells are observable and the structure of the grid is well defined, the MDP can be used in our approach.

Our overall approach is shown in Figure 1 with each number represents a major step. It can be summarized as follows:

Step 1) The smart device continuously collects the user's touch data during the smart device usage. Since most of the features we require are similar to those used in [6], we use the same procedure to extract the data from the smart device.

Step 2) The smart device analyzes the collected sensory data flows in *Step 1)* to generate the required features discussed in Section III. The output of this step is the features to be used for estimating importance of each cell in *Step 3).*

Step 3) Using the features generated in *Step 2)* to run algorithm discussed in Section IV on user's smart device for continuously updating the MDP state graph during the user's smart device usage session. The inputs for this algorithm include the features of user finger gestures on the smart device's touch screen, such as pressure, speed and usage frequency, as discussed in Section IV.

Step 4) Identify the current user's usage context of the smart device using the primary and secondary features generated in *Step 2), as* described in Section V.

Step 5) Generate the plans which are the sequences of cells in the MDP state graph representing the user's finger gestures, along with the plans' respective cumulative reward values based on the current user's usage context generated in *Step 4).* These plans are generated using the MDP algorithm as discussed in Section VI.

Step 6) The plans are used to continuously authenticate the legitimate user's macro-gestures in real time based on the cumulative reward values of the plans, as discussed in Section VI.

Our continuous authentication approach is flexible to the selection of number of cells and can be selected based on factors, such as type of user's smart device, screen size, degree of uniqueness of user gestures to be captured. It is also important to note that increasing the number of cells also increases the operational complexity. So an optimal trade-off is needed to be achieved based on the application specification, since the accuracy of the continuous authentication depends on the number of cells on touch screen of the smart device accurately. The details of the grid representation in the MDP will be explained in Section VI.

VIII. AN ILLUSTRATIVE EXAMPLE

Since *Step 5),* configuring the MDP, is the most difficult part, and has a major impact on the effectiveness of our approach. Hence in this section, we will illustrate the value iteration and policy iteration techniques of MDP using simulated reward values and probabilities. As shown in Figure 2, we have divided the touch screen of the smart phone into twelve cells. We simulated the reward value of each cell using (5) from the primary and secondary features described in Section III.

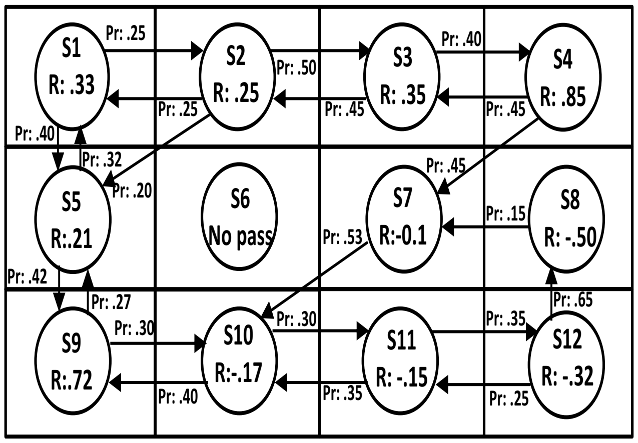

Figure 2: Simulated MDP model for the user finger gestures for this example

In Figure 2, based on the cell with high reward values we can observe that the states S1, S5, and S9 are more important as they are located on left column which is in the "unlock area" of touch screen of the smart device. Similarly, the states S1, S2, S3 and S4 are also heavily used because they are located top row which is in the "scroll area" of the touch screen. The reward values that are assigned to cells will vary on each individual user's usage pattern. A No Pass cell S6, is simulated in the MDP state graph to include scenarios where some parts of the touch screen are rarely used. Some of the cells have negative values which represent the minimal usage of those parts of the smart phone screen. Based on the simulated reward metrics, probabilities and MDP state graph of the touch screen, the value iteration technique and the policy iteration techniques were run and generated the plans along with their cumulative reward values.

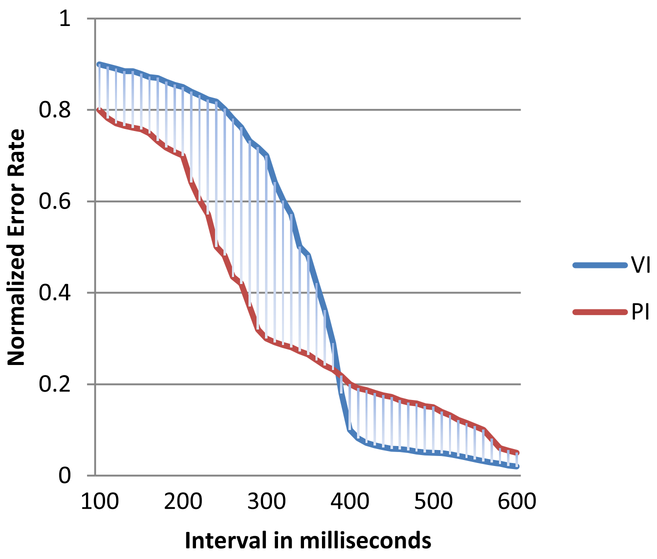

Figure 3: Comparison of the performance for value iteration and policy iteration techniques for this example

Figure 3 shows the performance of both the techniques of the MDP for the state graph shown in Figure 2. The value iteration technique represented by the blue line has higher initial error rate than the policy iteration technique represented by the red line, because the value iteration technique starts the Bellman backup procedure without any prior knowledge on the plans. Hence, the value iteration technique has higher initial error rate. The policy iteration technique starts with an initial plan, and hence has a lower initial error rate. In this example, we initialized the policy iteration technique with a suboptimal plan and hence the initial error rate is relatively low. The assignment of a suboptimal plan is valid because in the real world we would know some of the typical user finger gesture characteristics. Hence, we will be able to use user finger gesture characteristics for initial plan assignments.

In Figure 3, as both the techniques progress, the error rate of the policy iteration technique drops rapidly compared to the value iteration technique. This is because the policy iteration technique has additional prior knowledge on the user's finger gesture inputs. But, additional finger gesture inputs introduce biases into the policy iteration technique. Consequently, the policy iteration technique takes much longer time to find good plans than the value iteration technique. The blue line representing the value iteration technique also shows a sharp reduction in the error rates and shows a sharp turn as it reaches 400ms. This is because the value iteration technique has achieved relatively optimal plans. In Figure 3, both lines intersect at a point where the value iteration technique is closer to the optimal plans than the policy iteration technique because the value iteration technique has no bias compared to the policy iteration technique.

IX. CONCLUSION AND FUTURE WORK

In this paper, we have presented an approach that uses MDP and touch gestures of the user to continuously authenticate the user on a touch-screen smart device. This approach incorporates continuously learning, adaptation and validation of the user's finger gestures, and also estimation the usage context of the smart device. The major advantage of this approach is that it does not require the user to consciously provide inputs for training purposes and does the learning, adaptation and validation of the user's finger gestures in the background, invisible to the user. By continuously learning the user gestures, our system will be able to capture important unique user characteristics and also adapt to evolving user's finger gestures. This enables our system to build an accurate adaptive user gesture pattern that is very effective and efficient for continuous user authentication. Another major advantage of our approach is to have better authentication accuracy by treating uninterrupted user finger gestures over a short time interval as a single gesture for continuous user authentication

We need to conduct experiments to determine suitable weighting factor values for our approach. In addition, we plan to further reduce the memory and computational resources required for our approach by factorizing the cell attributes in MDP. We plan to integrate this approach with the development of smart IoT environment [18, 19]. We also plan to extend this approach to include the use of user-specific information, such as device type, user age, gender, geographic location, operating system to better incorporate user's characteristics.

X. REFERENCES

[1] Survey on the smart device shipment, "http://www.statista.com/statistics/259983/global-shipment-forecast-for-touch-screen-displays/", (press release in 2015) Accessed on May 15, 2015

[2] Framework for securing BYOD, "http://www.forescout.com/idc-fs-byod-strategy-white-paper/", (released on June, 2012) Accessed on May 15, 2015

[3] Wikipedia source on multi-factor authentication, "http://en.wikipedia.org/wiki/Multi-factor_authentication", Accessed on May 15, 2015

[4] Y. Niu, and C. Hao "Gesture authentication with touch input for mobile devices." *Security and Privacy in Mobile Information and Communication Systems*, Springer, Berlin Heidelberg, 2012, pp. 13-24.

[5] M. Frank, R. Biedert, E. Ma, I. Martinovic, and D. Song, "Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication.*", IEEE Trans. Information Forensics and Security,* (2013), Vol: 8(1), pp. 136-148.

[6] L. Li, X. Zhao, and G. Xue. "Unobservable reauthentication for smart phones", *Proc. the 20th Network and Distributed System Security Symposium*, (2013), Internet Society.

[7] J. Muncaster, and M. Turk. "Continuous multimodal authentication using dynamic Bayesian networks", *Proc. 2nd Workshop Multimodal User Authentication*, (2006).

[8] T. Sim, S. Zhang, R. Janakiraman, and S. Kumar. "Continuous verification using multimodal biometrics." *IEEE Trans. 29th Pattern Analysis and Machine Intelligence*, Vol: 4, (2007), pp: 687-700.

[9] W. Shi, J. Yang, Y. Jiang, F. Yang, and Y. Xiong. "Senguard: Passive user identification on smartphones using multiple sensors", *Proc. IEEE 7th International Conference on In Wireless and Mobile Computing (WiMob),* (2011), pp. 141-148.

[10] A. Serwadda, and V. V. Phoha. "When kids' toys breach mobile phone security." *Proc. ACM SIGSAC conference on Computer & communications security*, (2013), pp. 599-610.

[11] Y. Chen, and N. D. Thomas, "Active Learning of Markov Decision Processes for System Verification.", *Proc. 11th IEEE Conference on Machine Learning and Applications (ICMLA)*, 2012, vol. 2, pp. 289-294.

[12] K. Hamahata, T. Tadahiro, S. Kazutoshi, N. Ikuko, T. Kazuma, and S. Tetsuo, "Effective integration of imitation learning and reinforcement learning by generating internal reward.", *Proc. 8th IEEE Conference on In Intelligent Systems Design and Applications*, 2008, vol. 3, pp. 121-126.

[13] S. S. Yau, Y. Yin, and H. G. An, "An Adaptive Approach to Optimizing Tradeoff between Service Performance and Security in Service-based Systems", *International Journal of Web Services Research*, Vol. 8(2), 2011, pp. 74-91.

[14] S. Russell and P. Norvig, "*Artificial Intelligence: A Modern Approach"*, 3rd edition, Patience Hall (2009).

[15] R. A. Howard, "Dynamic Programming and Markov Processes", *MIT Press*, Cambridge, Massachusetts, 1960.

[16] R. Bellman, "Dynamic Programming", *Princeton University Press*, 1957.

[17] M. L. Puterman, "*Markov decision processes: Discrete stochastic dynamic programming*", John Wiley & Sons, 1994.

[18] S. S. Yau and A. Buduru, "Intelligent Planning for Developing Mobile IoT Applications Using Cloud Systems*", Proc. 3rd IEEE International Conference on Mobile Services (MS)*, June 2014, pp. 55-62

[19] S. I. Ahamed, I. Addo, S. S. Yau, A. Buduru, "A Reference Architecture for Improving Security and Privacy in Internet of Things Applications", *Proceedings of 3rd IEEE International Conference on Mobile Services (MS)*, June 2014, pp. 108-115