Mouse Dynamics Analysis Using Machine Learning to Prevent Account Stealing in Web Systems

Alexey Marakhtanov, Evgeny Parenchenkov, Nikolai Smirnov
Center of Artificial Intelligence, Petrozavodsk State University
Petrozavodsk, Russian Federation
{marahtanov, smirnovn}@petrsu.ru, parenche@cs.karelia.ru

***Abstract*—This paper is dedicated to development of algorithm of website users authentication based on their mouse movement dynamics. The algorithm of mouse activity data collection, feature extraction and mouse dynamics sessions classification to legal and illegal ones using machine learning and deep learning methods is proposed. The user authentication problem is considered as one-class classification and binary classification problem. For each user, the classifier that provides maximum session classification quality is automatically chosen. The performance of the algorithm is evaluated on mouse dynamics dataset gathered on a public website and compared with the results of the state-of-the-art solutions. The quality of mouse dynamics sessions classification using classifiers selected independently for each user is significantly higher than one when using the same session classifier for all system users.**

I. Introduction

Nowadays, as the number of online purchases is rapidly growing [1], the number of occasions of various types of fraud is also rising [2]. There are many types of fraud, from system intrusions to social engineering, but stealing of accounts seems to be the most popular and therefore the most dangerous fraud type in the field of e-commerce. The methods of confronting account stealing using machine learning and deep learning techniques are considered in this paper.

There are different means of user accounts protection. Usage only of login and password pair as the basic type of protection is proven an ineffective method because criminals can obtain such credentials either by simple brute-force algorithms or by analyzing compromised account databases.

There exist more complicated authentication methods that can protect the accounts more reliably. First, the two-factor authentication methods, when the user has to input the single-use code when logging into the system, are popular [3, 4]. Two-factor authentication's main limitation is that not all users agree to enable it, thus, their accounts remain vulnerable to basic attacks on credentials. Second, the biometry authentication methods, such as voice [5] or eye iris [6] analysis, gained popularity in recent years. They are proven to be secure, but have limited applicability as they require an explicit user's agreement to use his or her biometric data, and additional costly sensors sometimes have to be used for authentication.

Another class of authentication methods, which is discussed in this paper, is the behavior authentication methods. Such methods use information of how the user interacts with the system as input data to make a decision whether a person who is trying to log in is the true account owner or a fraudster. Behavior authentication methods do not distract the users from working with the system and allow executing the so-called continuous authentication: the user identity can be verified multiple times during one session in the system. These benefits make behavior authentication methods gain popularity in automated online environments.

Mouse dynamics analysis is one of the behavior authentication methods. Mouse dynamics means a way of how the user interacts with computer mouse or touchscreen. To use mouse dynamics for user authentication, first, the information about the user's interaction with computer mouse is gathered and stored as so-called user profile. Second, the information in the profile is compared with the user's future interactions with the system, and, if the new information differs from the profile significantly, the user is suspected to be a fraudster.

Although some research has already been conducted in the field of mouse dynamics authentication, there is no system that allows identifying fraudsters' intrusions perfectly, so development of novel approaches of mouse dynamics analysis is still a problem that needs further research.

The contributions of our research to the stated problem are:

- the technique of anonymous mouse dynamics data acquisition from the websites and extraction of large set of mouse dynamics features;
- the approach of mouse dynamics sessions classification to legal and illegal ones using machine learning algorithms and neural networks. Such an approach allows to automatically apply different algorithms to session classification problem;
- the algorithm of selecting the best session classifier for each system user; this algorithm allows to obtain the maximum possible session classification quality for each user and thus significantly improve the overall illegal session detection quality.

The rest of the paper is organized as follows. Section II describes the state-of-the-art solutions and summarizes their advantages and drawbacks. Section III describes the proposed user authentication algorithm, including data acquisition process, extracted mouse dynamics features, used classification methods and metrics and the proposed approach of mouse dynamics sessions classification. Section IV describes our experimental setup: data source, number of users

involved in the research and experimental system architecture. Section V contains the results of application of our algorithm to acquired dataset, Section VI compares our results with state-of-the-art solutions, and Section VII concludes the paper.

II. Related work

The research of behavior authentication can be conducted in two types of environment – controlled and uncontrolled one. Controlled environment means that the scenario of user behavior was pre-defined by the study authors. This environment is applicable only for research purposes as free user behavior is more diverse and stochastic than one defined by scenario. The interactions of users with real-life systems are studied in uncontrolled environment, and user authentication problem in uncontrolled environment is more challenging than one in controlled environment.

In [7], the authors compare performance of different vector-based classifiers, like Euclidean distance classifier or support vector machine, in the task of mouse dynamics sessions classification. This study is broad as it summarizes the applicability of multiple algorithms to mouse dynamics-based authentication, but it is conducted in controlled environment that restricts its applicability for free user actions analysis.

In [8], so-called patterns, regular sequences of actions that can characterize the users' behavioral habits, are discovered. The authors state that the mouse dynamics features extracted from patterns are more informative than ones extracted from continuous mouse dynamics sessions. The results of this research look promising as it involves large number of users that work in uncontrolled environment, and the declared error rate does not exceed 1%. Such a result significantly outperforms the results of other studies. However, the described method is evaluated on data of specific users. The method evaluation on other users' data is a point of interest. The results of this method application on dataset collected by us are given further.

Authors of [9] consider mouse movement sequence as a digital signal and apply Hilbert-Huang transform to extract features from it. Their approach significantly differs from most of ones proposed in other studies, but they involved only 10 participants in controlled environment for their experiments.

In [10], the authors suggest to combine mouse dynamics with keyboard activity features to increase the number of features available for authentication algorithm. They use multiple kernel learning technique to map keyboard and mouse features to different kernels suitable for each subset of features. The limitations of provided results are small number of research participants and short list of available keyboard features.

Authors of [11] propose a method of mouse dynamics sessions classification based on representation of sessions as images which show the cursor trajectory and points of screen where special events, such as scrolling or dragging, occur. They use convolutional neural networks as session images classifiers. They reached error rate less than 3%, but the results of method evaluation with larger number of users are not provided.

In [12], the authors compare performance of several machine learning algorithms (K nearest neighbors, decision trees and random forest) in the task of users' authentication by the information about mouse movements and point-and-click actions. They develop a large set of cursor trajectory features and reach high classification quality, but their study is limited by number of involved users – there were only 20 participants – and by the conditions of working process. The dataset used in their experiment was collected in a browser videogame.

Authors of [13] study the availability to combine information about mouse dynamics, keystrokes and HTTP context (browser type, IP address, etc.) to create a more robust authentication model than one that uses one feature set only. They combine user behavior information from TWOS dataset [14] and HTTP context data from a private banking web service dataset. They artificially create several intrusion scenarios. The authors state that training a model that simultaneously uses the result of behavioral-based and context-based authentication allows identifying the attacks on credentials more accurately than analyzing behavior or context data solely. The study is deep, but there are only 24 users in TWOS experiment, and the HTTP data is created synthetically.

The results of above-mentioned studies, including used machine learning methods, type of environment, number of participants and classification metrics, are shown in Table I.

TABLE I. Summary of results of related work

Source	Classification algorithms	Environment	Number of users	Classification quality metrics: FAR/ FRR / F_1
[7]	Euclidean distance, Manhattan distance, Mahalanobis distance, Outlier Counting, Nearest Neighbor, K Means, perceptron, one-class SVM	controlled	58	8.81% / 8.81% / -
[8]	One-class SVM, perceptron, K Nearest Neighbors	uncontrolled	159	0.09% / 1% / -
[9]	Bagged trees	controlled	10	- / - / 0.892
[10]	Decision Tree, Random Forest, Naïve Bayes, One-class SVM, SVM	uncontrolled	21	8.8% / 11.9% / -
[11]	Convolutional neural network	uncontrolled	10	2.94% / 2.28% / -
[12]	Decision Tree, K Nearest Neighbors, Random Forest, Convolutional neural network	controlled	20	0.052% / 0.99% / 0.957
[13]	Random Forest, SVM	uncontrolled	24	- / - / 0.915

TABLE II. MOUSE EVENT STRUCTURE

Field name	Field description
ts	Timestamp – number of milliseconds passed from session start
button	Mouse button code: 1 – no button, 2 – left button, 3 – right button, 4 – middle button
state	Event numerical code: 1 – mouse cursor movement, 2 – mouse button is released, 3 – scroll up, 4 – drag-and-drop, 5 – mouse button 6 – scroll down
x	Abscissa of the cursor position relative to the upper-left edge of the page
y	Ordinate of the cursor position relative to the upper left edge of the page
session_id	Unique session identifier

III. ALGORITHM OF USER AUTHENTICATION BY MOUSE DYNAMICS

Our purpose is to create the classifiers based on machine learning and deep learning methods that get the information about the mouse dynamics of user during the session and return the value of probability that the user working with the system is a fraudster. To do so, we have to perform the following steps:

- gather the information about the user's mouse dynamics activity,
- extract informative features from raw mouse events data,
- train mouse sessions classifiers based on machine learning methods.

A. Mouse dynamics data collection

In order to obtain the information about mouse events occurring during the session of user interaction with the web system, a script that can handle the events of mouse activity should be developed and added to this system. This script should be able to collect the information about such events as mouse movements, left and right clicks, double clicks, scrolling and drag-and-drop. Information about mouse events generated by users should be grouped into so-called sessions.

The proposed structure of information about single mouse event is described in Table II. The matching of events and their numerical identifiers is based on public Balabit mouse dynamics dataset [15].

B. Mouse dynamics feature extraction

Only coordinates and types of events obtained by the mouse event detection script are not enough to train high-quality machine learning models, so several groups of features that describe the user's behavior more precisely can be extracted from the event sequences.

Feature extraction techniques used in the research are based on approaches [16] and [17] for mouse movement features and click features extraction respectively.

The authors of [16] define a stroke as a sequence of mouse movements between two consecutive clicks. However, such definition ignores the fact that the user may be inactive during the session, that is, he or she does not move the mouse for some time. In order to consider that the movement trajectory may contain pauses, we define the threshold value between two consecutive timestamps.

$$t_{i+1} - t_i \geq \Delta_1 \quad (1)$$

t_{i+1} is the i+1-th event timestamp, t_i is the i-th event timestamp in the same session, Δ_1 is the inter-stroke time threshold value.

If (1) is true for two consecutive cursor movements, then the current stroke is considered completed. In addition, any stroke should contain not less than 3 points of trajectory.

After dividing the mouse trajectory into the strokes, values of several features are calculated to characterize each stroke based on information about its points. There are three groups of stroke features:

- Temporal features (defined in [16]) that use timestamps of mouse events; this group of features includes X-axis offset, Y-axis offset, X-axis velocity, Y-axis velocity, tangential velocity, acceleration, tangential jerk, angular velocity, last action time (mentioned as time to click in [16]), number of pauses, paused time and ratio of paused time and duration of the whole session. For X-axis offset, Y-axis offset, X-axis velocity, Y-axis velocity, tangential velocity, acceleration, tangential jerk and angular velocity, their minimum, maximum, average value, standard deviation and range across the stroke are used as resulting features.
- Spatial features (defined in [16]) that use only coordinates of trajectory points in the stroke; this group of features includes the angle θ between the tangent to the trajectory and X-axis, the path traveled at the last point of the stroke (in pixels), the trajectory curvature and curvature change ratio, trajectory straightness and number of critical points. For angle θ, curvature and curvature change ratio, their minimum, maximum, average value, standard deviation and range across the stroke are used as resulting features.
- Beginning of action (BoA) and finishing of action (FoA) (defined in [17]) are the parts of the stroke trajectory where the acceleration is non-negative and non-positive respectively. It should be noted that users move the mouse cursor in non-uniform way, so there may be several BoA and FoA points during the stroke. In such situation, the metrics are calculated for the first pair of BoA and FoA of the stroke. These metrics are minimum, maximum, average value, standard deviation, range of acceleration values sequence during BoA and FoA; the timestamps of BoA and FoA are also saved as features.

We treat the scrolling events as mouse movement, so the set of features similar with one described above is extracted for scrolling events.

The information about mouse clicks is processed apart from the movement information. Each click is composed from two consecutive events – button press and button release. In order to distinguish double clicks from single clicks, the threshold value Δ_2 between timestamps of the first button release (t_{r_1}) and the second button press (t_{p_2}) was defined.

$$t_{p_2} - t_{r_1} \leq \Delta_2 \quad (2)$$

If (2) is true for the couple of consecutive clicks, then this couple is treated as a double click, as a single click otherwise.

The features of the clicks are introduced in [17]. They included:

- for single click: the duration of the left click, the duration of the right click;
- for double click: the first click time, the second click time, the total time of double click, the time between button presses and the time between button releases.

Totally 77 features were calculated for an event (stroke or click) in the final representation of user's mouse dynamics session. For a stroke, the click features are replaced with zeros and for a click vice versa.

C. Used machine learning and deep learning methods

We consider the task of classification of the mouse dynamics sessions to legal ones (performed by actual owner of the account) and illegal ones (performed by some other user) in terms of two types of machine learning problems – as one-class classification or binary classification problem. Solving multiclass classification problem to identify the exact user who performed the mouse dynamics session is not considered in this paper as it becomes difficult for large number of users because there will be not enough data to describe each user precisely.

The main feature of one-class machine learning methods is that they are trained using only the objects that belong to the single class; the objects of other class are treated as outliers or anomalies, depending on the used algorithm.

We use the following one-class classification algorithms:

- One-class support vector machine (One-class SVM) is a modification of traditional SVMs designed for outlier detection. It divides the feature space by a hyperplane to constrain the region where the training data is the most concentrated, and all the objects that are outside this hyperplane are labeled as outliers.
- Isolation forest is a one-class modification of random forest that isolates the objects by randomly selecting its features and randomly selecting a split value between the maximum and minimum values of the selected feature. The path length from the tree root node to the node where the splitting is terminated is a metric of the object abnormality. If the average path for this object among the forest trees is short, then the object is treated as an outlier.
- The local outlier factor (LOF) algorithm computes the local density deviation of the current object from its neighbors in the feature space. If the object has significantly lower density than its neighbors do, then it is treated as an outlier.

We use the implementations of one-class SVM, isolation forest and LOF from Scikit-Learn [18] Python library.

The task of account stealing detection can also be considered as a binary classification problem when the sessions belonging to the actual account owner are treated as objects of the negative class and the sessions of all other users form the set of positive class objects. Applying this method, we have a larger training set than one for one-class classification, but the training time increases respectively.

We chose perceptron as a classifier for our problem. Perceptron is one of the basic neural networks that consists of input layer, one or several fully connected hidden layers and one output layer. During our experiments, we discovered that one hidden layer with 32 neurons and Swish activation function [19] is enough for our network to provide high classification quality.

To construct the perceptron, the implementations of neural network components from Keras [20] Python library are used.

D. Used classification metrics

To evaluate the classification quality of trained models, we use such metrics as precision (3), recall (4), F_1 score (5), ROC AUC, false acceptance rate (FAR) (6) and false rejection rate (FRR) (7).

$$\boldsymbol{precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP + FP}} \quad (3)$$

$$\boldsymbol{recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP + FN}} \quad (4)$$

$$\boldsymbol{F_1} = \frac{\boldsymbol{2}}{\boldsymbol{precision^{-1} + recall^{-1}}} \quad (5)$$

$$\boldsymbol{FAR} = \frac{\boldsymbol{FN}}{\boldsymbol{FN + TP}} \quad (6)$$

$$\boldsymbol{FRR} = \frac{\boldsymbol{FP}}{\boldsymbol{FP + TN}} \quad (7)$$

TP is the number of true positives (number of correctly classified illegal sessions), *TN* is the number of true negatives (number of correctly classified legal sessions), *FP* is the number of false positives (number of legal sessions misclassified as illegal), *FN* is the number of false negatives (number of illegal sessions misclassified as legal).

Area under receiver operational characteristic curve (ROC AUC) is drawn in the coordinates of true positive rate (TPR, the fraction of correctly classified objects of positive class) and false positive rate (FPR, the fraction of negative class

objects that are misclassified as positives). ROC AUC allows estimating the model's classification quality independently of the selected positive class probability threshold. The closer ROC AUC to 1, the higher classification quality the model provides.

E. Mouse dynamics sessions classification algorithm

As it is mentioned in Section III.C, the negative class is formed from the events of the sessions performed by the actual account owner, and the events of all other users' sessions form the positive class. Thus, the division of events to classes is different for each unique user. Consequently, an individual classifier corresponds to each user in our algorithm.

The training and testing sets for current user are formed as follows:

- The training set for one-class classification algorithms contains events from all sessions of the current user, except the events from session that has the latest timestamp; for perceptron, the events from sessions of all other users, except the latest ones, are added as the objects of positive class to the dataset with the events of the current user.
- The testing set for both one-class algorithms and perceptron contains the events of the latest session of the current user (as negative class session) and the events of the latest sessions of all other users (as positive class sessions).

The output probability of session being legal is calculated by (8):

$$\boldsymbol{P(s \in u) = \frac{\sum_{i=0}^{n} P(e_i \in u)}{n}} \quad (8)$$

$P(s \in u)$ is the probability of the session s being performed by the current user u, n is the number of events in the session s, $P(e_i \in u)$ is the probability of the event e_i being performed by the user u, which is the output of the classifier.

The class label is assigned to the session according to the rule (9):

$$\boldsymbol{y(s) = \begin{cases} 0, P(s \in u) \geq \lambda \\ 1, otherwise \end{cases}} \quad (9)$$

$y(s)$ is the label of the session s (0 if the session belongs to the user u, 1 otherwise), $P(s \in u)$ is the probability of the session s being performed by the current user u which is calculated by (6), λ is the threshold probability of classifying the session as the legal one.

For each user, all four algorithms discussed in the Section III.C are trained, and the model that provides the highest values of F_1 score and ROC AUC is saved as the classifier corresponding to the current user. The process of obtaining of the models is as follows:

- for each user, the λ threshold is iterated on (0; 1) interval with step of 0.005;
- for each λ, four classifiers are trained and tested, the classification metrics values are obtained;
- if F_1 score and ROC AUC of new classifier are higher than the previous highest scores for the current user, new classifier is set as the best classifier for that user.

F. Authentication time estimation

One of advantages of behavior authentication that is also a subject of our analysis is that it does not require additional actions from the system user, thus, it can be performed repeatedly during the user's activity session. We conducted an experiment with trained classifiers to estimate the period of time that can be used for reliable user re-authentication.

The experiment was arranged as follows: for $n=1, ..., 12$ we passed the information about mouse dynamics events that occurred during first $n*10$ seconds of the test sessions to the trained classifiers and measured the average F_1 score among the classifiers of all users.

IV. Experimental setup

We selected the PetrSU Portfolio [21] website as a platform for data collection as students and lecturers actively use it to view schedule and homework and share other information. The process of data gathering was anonymous: for each user, we created a unique identifier that is impossible to associate with the real user. Our dataset was obtained in uncontrolled environment as the users performed their daily tasks without following any pre-defined scenarios.

To collect the information about mouse events, we developed the JavaScript plugin described in Section III.A and placed it on different pages of Portfolio. Information about mouse events generated by users was grouped into sessions: each time the user moved to other Portfolio page, new session was initiated. Each session had a unique identifier that was generated with salted MD5.

Each event was represented as a string which contained fields described in Table II separated by comma. The sequences of events separated by semicolon were sent to the web server every 5 seconds during the session.

The web server that received requests from plugin was created using Django framework for Python programming language. This server converted strings to numerical values and sent information about the events that was obtained by the plugin to MySQL database. The scripts that generate the mouse dynamics features and train the session classifiers were also written in Python. The values of Δ_1 and Δ_2 were set to 250 ms and 300 ms respectively.

Information about mouse dynamics of Portfolio users was being collected during 5 months. 13.3 million events were registered during the research period. The number of events generated by different users ranged significantly, so for the algorithm evaluation among 7638 unique users of Portfolio who logged in the website during the experiment period we picked the events produced by 100 users who generated maximum number of events.

Drag-and-drop events occurred rarely: only 340 thousand of events among 13.3 million events in the database corresponded to drag-and-drop (less than 3%), so we decided

to remove them from the event sequences while extracting the mouse dynamics features.

Fig. 1 shows the pipeline of experimental system that was created during the research.

V. Algorithm evaluation results

Table III contains classification metrics averaged among all 100 users, data of whose mouse dynamics were included into the experimental dataset.

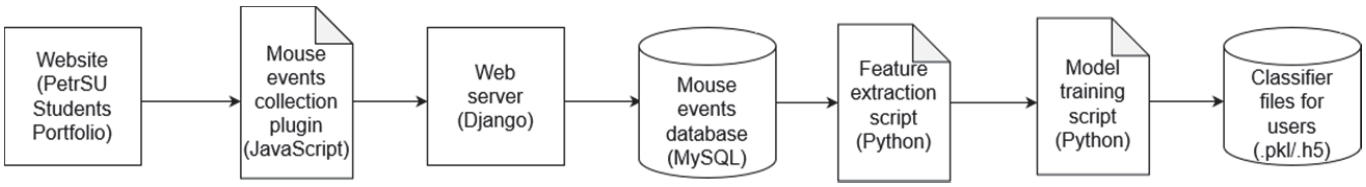

Fig. 1. Experimental pipeline of mouse dynamics authentication system

The significant difference between the metrics obtained with four individual machine learning algorithms and the best models in Table III appears because for some users one-class SVM showed the best quality, but for others perceptron performed better. Isolation forest and LOF failed to classify mouse dynamics sessions with satisfying quality.

Fig. 2 shows that we can reach high performance ($F_1 > 0.8$) using the events of the first 30 seconds of mouse dynamics sessions, so we can re-authenticate the system users every 30 seconds in order to detect the fraudster's intrusion during the session.

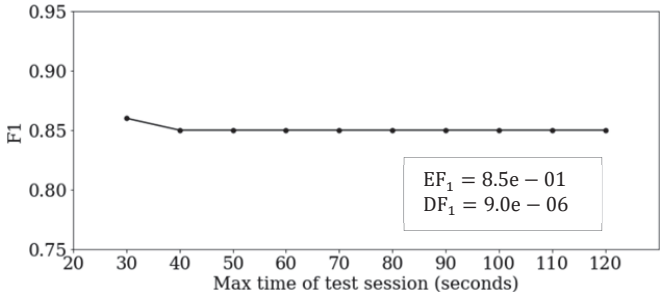

Fig. 2. Dependency of the average F_1 score on the duration of the user session

TABLE III. Average classification quality metrics for 100 Portfolio users mouse dynamics dataset

Algorithm	Precision	Recall	F_1	ROC AUC	FAR	FRR
One-class SVM	0.99	0.58	0.68	0.74	41.7%	10%
Isolation forest	0.96	0.06	0.11	0.5	95.2%	1%
LOF	0.99	0.21	0.35	0.52	79.2%	0%
Perceptron	0.99	0.88	0.92	0.6	11.6%	10%
Best classifiers	0.99	0.87	0.92	0.93	12.5%	2%

Fig. 2 shows the graph of dependency between the duration of the input sessions and the average F_1 score of all users' classifiers. *n=1, 2* were discarded as there are sessions that do not have any events during the first 20 seconds in the dataset. The values of the average F_1 score obtained for each value of *n* are marked with black dots on the chart.

The proposed algorithm of selecting the best individual model for each system user allowed classifying the mouse dynamics sessions to legal and illegal ones efficiently (average $F_1 > 0.9$ and $AUC > 0.9$).

VI. Discussion

The proposed algorithm looks promising as it provides precise mouse dynamics session classification, and its quality is comparable to existing studies. The results of [7] and [12] outperform ours due to their controlled environments and smaller number of research participants. Our algorithm shows comparable values of metrics with results of [10] and [13], but our dataset contains significantly larger number of participants.

The solutions suggested in [9] and [11], although their stated classification quality is high, have a significant limitation compared to our algorithm as they both are evaluated on small groups of 10 users each.

As it is mentioned in Section II, the session classification quality stated in [8] is very high, so we conducted an experiment to compare the performance of our mouse dynamics session classification quality and theirs. We reproduced the pattern extraction algorithm precisely following the steps described in [8] and applied it to our dataset. The F_1 score achieved with the pattern-based session classification algorithm was 0.88, whereas classifiers trained with our algorithm allowed to reach F_1 = 0.92 (see Table III). Thus, we claim that our mouse dynamics sessions classification approach shows higher quality than one suggested in [8] for some datasets.

VII. Conclusion

The algorithm of authentication the web system users by information about their mouse dynamics is proposed. It combines one-class and binary classification and allows creating individual mouse dynamics classifier for each system user. First, the technique of unobtrusive mouse dynamics data acquisition using JavaScript plugin that can be embedded on any website is designed. Second, the approach of extracting of 77 mouse dynamics features from data collected by plugin is proposed. Finally, we designed the algorithm of mouse dynamics sessions classification to legal and illegal ones that

allows to automatically select the classifiers that provide highest session classification quality for each system user. The system provides session classification quality comparable to the state-of-the-art solutions (F_1 score = 0.92, ROC AUC = 0.93, FAR = 12.5%, FRR = 2%). The classification quality provided by the best classifiers is significantly higher than one provided by single algorithm for all system users in terms of F_1 score, ROC AUC, FAR and FRR.

In future, the designed algorithm can be used for continuous user authentication in the web systems: the mouse dynamics features of data obtained by the front-end plugin can be extracted and passed to the session classifier directly on the web server, and the suspicious sessions may be blocked before the fraudster finishes the malicious work.

ACKNOWLEDGEMENT

The authors are grateful to anonymous reviewers of the FRUCT Association and experts of Center of Artificial Intelligence of PetrSU for their valuable comments to improve the quality of this paper.

This research is implemented in Petrozavodsk State University (PetrSU) with financial support by the Ministry of Science and Higher Education of Russia within Agreement no. 075-15-2021-1007 on the topic "Software and hardware methods of sensorics and machine perception for robotic systems with autonomous movement".

REFERENCES

[1] J. Verdon, Global e-commerce sales to hit $4.2 trillion as online surge continues, Adobe reports, Web: https://www.forbes.com/sites/joanverdon/2021/04/27/global-ecommerce-sales-to-hit-42-trillion-as-online-surge-continues-adobe-reports.

[2] LexisNexis Risk Solutions official website, Fraud trends to watch in 2021, Web: https://risk.lexisnexis.com/insights-resources/infographic/fraud-trends-to-watch-in-2021.

[3] M.J. Hossain, C. Xu, C. Li, S.M.H. Mahmud, X. Zhang, W. Li, "ICAS: Two-factor identity-concealed authentication scheme for remote-servers", *Journal of Systems Architecture,* vol.117, Aug.2021.

[4] E. Rushdy, W. Khedr and N. Salah, "Framework to secure the OAuth 2.0 and JSON web token for rest API", *Journal of Theoretical and Applied Information Technology,* vol.99, iss.9, May 2021, pp. 2144-2161.

[5] V.M. Antonova, K.A. Balakin, N.A. Grechishkina and N.A. Kuznetsov, "Development of an authentication system using voice verification", *Journal of Communications Technology and Electronics,* vol.65, iss.12, Dec.2020, pp. 1460-1468.

[6] M.K. Morampudi, M.V.N.K. Prasad, M. Verma and U.S.N. Raju, "Secure and verifiable iris authentication system using fully homomorphic encryption", *Computers and Electrical Engineering*, vol.89, Jan.2021.

[7] C. Shen, Z. Cai, X. Guan, R. Maxion, "Performance evaluation of anomaly-detection algorithms for mouse dynamics", *Computers & Security*, vol.45, Sep.2014, pp. 156-171.

[8] C. Shen, Y. Chen, X. Guan an R. Maxion, "Pattern-growth based mining mouse-interaction behavior for an active user authentication system", IEEE Transactions on Dependable and Secure Computing, vol.17, iss.2, Mar.2020, pp. 335-349.

[9] Y.G. Zhang, S.Q. Xiong, J.J. Li and S.P. Yi, "Use mouse ballistic movement for user authentication based on Huang-Hilbert transform", *Advances in Intelligent Systems and Computing*, vol.1219 AISC, Jul.2020, pp. 64-71.

[10] X. Wang, Q. Zheng, K. Zheng and T. Wu, "User Authentication Method Based on MKL for Keystroke and Mouse Behavioral Feature Fusion", *Security and Communication Networks,* vol.2020, 2020.

[11] T. Hu, W. Niu, X. Zhang, X. Liu, J. Lu, Y. Liu, "An Insider Threat Detection Approach Based on Mouse Dynamics and Deep Learning", *Security and Communication Networks,* vol.2019, 2019.

[12] S. Almalki, N. Assery, K. Roy, "An empirical evaluation of online continuous authentication and anomaly detection using mouse clickstream data analysis", *Applied Sciences (Switzerland),* vol.11, iss.13, Jul.2021.

[13] J. Solano, L. Camacho, A. Correa, C. Deiro, J. Vargas, M. Ochoa, "Combining behavioral biometrics and session context analytics to enhance risk-based static authentication in web applications", *International Journal of Information Security*, vol.20, iss.2, Apr.2021, pp. 181-197.

[14] A. Harilal, F. Toffalini, I. Homoliak, J. Castellanos, J. Guarnizo, S. Mondal, M. Ochoa, "The Wolf of SUTD (TWOS): A dataset of malicious insider threat behavior based on a gamified competition", *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications,* vol.9, iss.1, Mar.2018, pp. 54-85.

[15] GitHub, balabit/Mouse-Dynamic-Challenge, Web: https://github.com/balabit/Mouse-Dynamics-Challenge.

[16] H. Gamboa, A. Fred, "A behavioural biometric system based on human computer interaction", *Proceedings of SPIE – The International Society for Optical Engineering*, vol.5404, Apr.2004, pp. 381-392.

[17] M. Yildirim, E. Anarim, "Novel feature extraction methods for authentication via mouse dynamics with semi-supervised learning", *Proceedings – 2019 Innovations in Intelligent Systems and Applications Conference, ASYU 2019*, Oct.2019.

[18] scikit-learn: Machine learning in Python, 2.7. Novelty and Outlier Detection, Web: https://scikit-learn.org/stable/modules/outlier_detection.html.

[19] P. Ramachandran, B. Zoph, Q.V. Le, "Searching for activation functions", *6th International Conference on Learning Representations, ICLR 2018 - Workshop Track Proceedings*, May 2018.

[20] Keras: the Python deep learning API, Web: https://keras.io.

[21] PetrSU students portfolio, Web: https://portfolio.petrsu.ru.