Using Closed Itemsets for Implicit User Authentication in Web Browsing

O. Coupelon[1], D. Dia[1,2], F. Labernia[2], Y. Loiseau[2], and O. Raynaud[2]

[1] Almerys, 46 rue du Ressort, 63967 Clermont-Ferrand
{olivier.coupelon,diye.dia}@almerys.com
[2] Blaise Pascal University, 24 Avenue des Landais, 63170 Aubière
{loiseau,raynaud}@isima.fr, fabien.labernia@gmail.com

Abstract. Faced with both identity theft and the theft of means of authentication, users of digital services are starting to look rather suspiciously at online systems. The behavior is made up of a series of observable actions of an Internet user and, taken as a whole, the most frequent of these actions amount to habit. Habit and reputation offer ways of recognizing the user. The introduction of an implicit means of authentication based upon the user's behavior allows web sites and businesses to rationalize the risks they take when authorizing access to critical functionalities. In this paper, we propose a new model for implicit authentication of web users based on extraction of closed patterns. On a data set of web navigation connection logs of 3,000 users over a six-month period we follow the experimental protocol described in [1] to compute performance of our model.

1 Introduction

In order to achieve productivity gains, companies are encouraging their customers to access their services via the Internet. It is accepted that on-line services are more immediate and more user-friendly than accessing these services via a brick and mortar agency, which involves going there and, more often than not, waiting around [2]. Nevertheless, access to these services does pose security problems. Certain services provide access to sensitive data such as banking data, for which it is absolutely essential to authenticate the users concerned. However identity thefts are becoming more and more numerous [3]. We can distinguish two paradigms for increasing access security. The first one consists of making access protocols stronger by relying, for example, on external devices for transmitting access codes that are supplementary to the login/password pair. Nevertheless, these processes are detrimental to the user-friendliness and usability of the services. The number of transactions abandoned before reaching the end of the process is increasing and exchange volumes are decreasing. The second paradigm consists to the contrary of simplifying the identification processes in order to increase the exchange volumes. By way of examples, we can mention single-click payment [2] [4] or using RFID chips for contactless payments. Where these two paradigms meet is where we find implicit means of authentication.

A means of authentication is a process that makes it possible to ensure that the identity declared in the event of access is indeed the user's identity. Traditionally, a user authenticates himself or herself by providing proof of identity [5]. This process is called "explicit authentication". In contrast, implicit authentication does not require anything from the user but instead studies his or her behavior, the trail left by the user's actions, and then either does or does not validate the declared identity. An implicit means of authentication cannot replace traditional means of authentication as it is necessary for the user to have access to his or her service so that the person's behavior may be studied and their identity can either be validated or rejected. To the contrary, if it is effective, it would enable stronger authentication modes to be avoided (such as chip cards and PIN numbers), which are detrimental to the usability of services. The challenge is to detect identity theft as quickly as possible and, to the contrary, to validate a legitimate identity for as long a time as possible.

This contribution is organized as follows: in section 2 we shall offer a state-of-the-art about implicit authentication and user's profile in web browsing. Then we propose a learning model for implicit authentication of web users we are dealing with in section 3. In section 4, we compare several methods for building profiles of each user. We faithfully reproduce the experimental study conducted in [1] and we analyze all of our results. Finally, in section 5, we shall resume our results and discuss our future work.

2 Related works

In his survey of implicit authentication for mobile devices ([6]), the author says of an authentication system that it is *implicit* if the system does not make demands of the user (see Table 1).

Implicit authentication systems were studied very quickly for mobile phones. In [7] and [8], the authors studied behaviour based on variables specific to smartphones such as calls, SMS's, browsing between applications, location, and the time of day. Experiments were conducted based on the data for 50 users over a period of 12 days. The data were gathered using an application installed by users who were volunteers. The users' profiles were built up from how frequently positive or negative events occurred and the location. Within this context, a positive event is an event consistent with the information gathered upstream. By way of an example, calling a number which is in the phone's directory is a positive event. The results of this study show that based on ten or so actions, you can detect fraudulent use of a smartphone with an accuracy of 95%. In a quite different context, the authors of [9] relied on a Bayesian classification in order to associate a behaviour class with each video streaming user. The data set is simulated and consists of 1,000 users over 100 days. The variables taken into account are the quality of the flow, the type of program, the duration of the session, the type of user, and the popularity of the video. The results are mixed, because the model proposed admits to an accuracy rate of 50%.

Feature	*Capturing Method*	*Implicit/Explicit*	*Spoofing Threats*	*Problems*
Passcode	Keyboard input	Explicit	Keyloggers, Shoulder Surfing	Guessable passwords
Token	Hardware device	Mainly explicit, implicit possible	None	Easily stolen or lost
Face & Iris	Camera	Both	Picture of the legitimate user	Lighting situation and make-up
Keystroke	Keyboard	Implicit, explicit possible	Typing imitation (difficult)	Long training phase, reliability
Location	GPS, infrastructure	Implicit	Informed strangers	Traveling, precision
Network	Software protocol (e.g. WireShark)	Implicit	Informed strangers	Precision

Table 1. Comparison of different authentication methods

The particular context of implicit authentication for web browsing was studied in [1], [10], [11] and [12]. In [1], the author adopted the domain name, the number of pages viewed, the session start time, and its duration, as characteristic variables. The data set, which was gathered by a service provider, consisted of 300 first connections by 2,798 users over a period of 12 months. The user profiles consisted of patterns with a size of 1. The author compares several pattern selection approaches like the support and the lift approaches. The study shows that for small, anonymous behavioural patterns (involving up to twenty or so sites visited), the most effective models are still traditional classification models like decision trees. On the other hand, whenever anonymous behaviour exceeds 70 or so sites, the support and lift-based classification models are more accurate. The study conducted in [12] states that the size of the data set remains a determining parameter. Their study, conducted on 10 users over a one-month period, did not enable them to build a significant model for distinguishing users. The authors also concluded that no variable taken individually enables a user to be authenticated. Drawing inspiration from a study conducted in [1], the authors of [13] studied several techniques for spying on a user who holds a dynamic IP address, based on behavioural models. The methods compared are seeking motives, the nearest neighbours technique, and the multinomial Bayesian classifier. The data set consisted of DNS requests from 3,600 users over a two-month period. In this study, only the most significant variables and the most popular host names were considered. The accuracy rates for the models proposed were satisfactory.

The study that we conduct in this paper also forms part of a continuation of the work by [1]. We faithfully reproduce his experimental protocol on our data and we compare performance of our classification algorithm to his specific models.

3 Models

We propose an intuitive learning model architecture for user authentication. From a data set of web browsing logs we compute a set of *own patterns* for each user. A pattern is a set of frequently visited sites. The size of pattern may vary. Thanks to these *profiles* we are able to provide an authentication for anonymous sessions. We then compute confusion matrices and we provide precisions of the models. In our present study, we compare performance of a naive Bayes classifier to variations on k-nearest neighbors algorithms. More precisely, the studied parameters are selection process of user *own patterns*, computation process of *user profiles* and distance functions computed for classification stage. Figure 1 outlines the framework of the machine learning process.

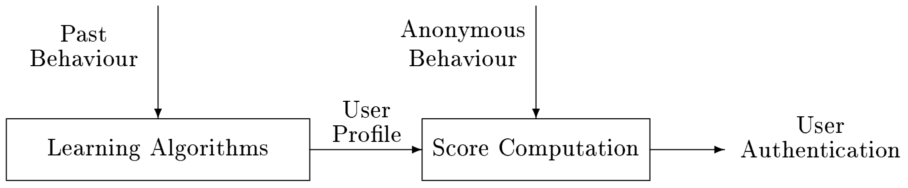

Fig. 1. Architecture

3.1 Formal framework

We call a session a set of visited web sites at a specific time by a given user u_i such as $i \in [1,n]$ and n is the number of users. The size of a session is limited and equal to 10. The learning database of each user u_i takes the form of a set of sessions denoted $\mathcal{S}_{u_i}$ and is built from log data[3]. We call $\mathcal{S} = \bigcup_i \mathcal{S}_{u_i}$ the whole set of sessions of the database.
We call W_{u_i} the whole set of web sites visited at least once by user u_i and we call $W = \bigcup_i W_{u_i}$ the whole set of visited sites. The order of visited web sites is not taken into account by this model.

Definition 1 (k-pattern). *Let W be a set of visited web sites and $\mathcal{S}$ be a set of sessions on W. A subset P of W is called a $k-pattern$ where k is the size of P. A session S in $\mathcal{S}$ is said to contain a $k-pattern$ P if $P \subseteq S$.*

Definition 2 (Support and relative support (lift)). *We define the support of a pattern P as the percentage of sessions in $\mathcal{S}$ containing P (by extension we give the support of a pattern in the set of sessions of a given user u_i):*

$$support_{\mathcal{S}}(P) = \frac{||\{S \in \mathcal{S} \mid P \subseteq S\}||}{||\mathcal{S}||} \qquad support_{\mathcal{S}_{u_i}}(P) = \frac{||\{S \in \mathcal{S}_{u_i} \mid P \subseteq S\}||}{||\mathcal{S}_{u_i}||}$$

[3] Cf. section 4.1

For a given user the relative strength of a pattern is equivalent to the $lift$ in a context of association rules (i.e. the support of the pattern within this user divided by the support of the pattern across all users). More formally:

$$lift_{\mathcal{S}_{u_i}\mathcal{S}}(P) = \frac{support_{\mathcal{S}_{u_i}}(P)}{support_{\mathcal{S}}(P)}$$

The *support* measures the strength of a pattern in behavioral description of a given user. The *relative support* mitigates support measure by considering the pattern's support on the whole sessions set. The stronger the global support of a pattern, the lesser characteristic of a specific user.
The tf-idf is a numerical statistic that is intended to reflect how relevant a word is to a document in a corpus. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the whole corpus ([14]). In our context, a word becomes a pattern, a document becomes a set of sessions $\mathcal{S}_{u_i}$ of a given user and the corpus becomes the whole set $\mathcal{S}$ of all sessions.

Definition 3 ($tf \times idf$). *Let P be a pattern, let U be a set of users and $U_p \subseteq U$ such that $\forall u_i \in U_p$, $support_{\mathcal{S}_{u_i}}(P) \neq 0$. Let $\mathcal{S}_{u_i}$ be a set of sessions of a given user u_i and $\mathcal{S}$ a whole set of sessions. The normalized term frequency denoted $tf(P)$ is equal to $support_{\mathcal{S}_{u_i}}(P)$ and the inverse document frequency denoted $idf(P)$ is equal to $log\ (||U||/||U_P||)$. We have:*

$$tf \times idf(P) = support_{\mathcal{S}_{u_i}}(P) \times log\left(\frac{||U||}{||U_P||}\right)$$

Definition 4 (Closure system). *Let $\mathcal{S}$ be a collection of sessions on the set W of web sites. We denote $\mathcal{S}^c$ the closure under intersection of $\mathcal{S}$. By adding W in $\mathcal{S}^c$, $\mathcal{S}^c$ is called a closure system.*

Definition 5 (Closure operator). *Let W be a set, a map $C\colon 2^W \rightarrow 2^W$ is a closure operator on W if for all sets A and B in W we have: $A \subseteq C(A)$, $A \subseteq B \Longrightarrow C(A) \subseteq C(B)$ and $C(C(A)) = C(A)$.*

Theorem 1. *Let $\mathcal{S}^c$ be a closure system on W. Then the map $C_{\mathcal{S}^c}$ defined on 2^W by $\forall A \in 2^W$, $C_{\mathcal{S}^c}(A) = \bigcap\{S \in \mathcal{S}^c \mid A \subseteq S\}$ is a closure operator on W [4].*

Definition 6 (Closed pattern[5]). *Let $\mathcal{S}^c$ be a closure system on W and $C_{\mathcal{S}^c}$ its corresponding closure operator. Let P be a pattern (i.e. a set of visited sites), we said that P is a closed pattern if $C_{\mathcal{S}^c}(P) = P$.*

[4] Refer to the book of [15].
[5] This definition is equivalent to a concept of the formal context $\mathbb{K} = (S,W,I)$ where S is a set of objects, W a set of attributes and I a binary relation between S and W [16].

3.2 Own patterns selection

The first and most important step of our model, called *own patterns selection* is to calculate the set of own patterns for each user u_i. This set of patterns is denoted $\mathcal{P}_{u_i} = \{P_{i,1}, P_{i,2},..., P_{i,p}\}$. In [1], the author states that $p = 10$ should be a reference value and that beyond this value model performance are stable. We shall follow that recommendation. In [1], 10 frequent $1-patterns$ are selected for each user. The aim of our study is to show that it could be more efficient to select closed $k-patterns$. But, the number of closed patterns should be strong, so we compare three heuristics (H_1, H_2 and H_3) to select the 10 closed patterns of each user. For each heuristic, closed patterns are computed thanks to Charm algorithm ([17]) provided on the Coron platform ([18]). Only closed patterns with a size lower than or equal to 7 are considered. These heuristics are presented here:

1. 10 $1-patterns$ with the largest *support* values (as in [1])
2. H_1: 40 closed $k-patterns$ with the largest *tf-idf* values.
3. H_2: 10 filtered closed $k-patterns$ with the largest *support* and maximal values by inclusion set operator.
4. H_3: 10 filtered closed $k-patterns$ with the largest *tf-idf* and minimal values by inclusion set operator.

Algorithm 1 describes the process of H_1 to select the 40 own patterns for a given user. With H_1, the model performance is improved when p increases up to 40. $p = 10$ is the better choice for H_2 and H_3. The best results are from H_1.

Algorithm 1: H_1: 40 closed $k-patterns$ with the largest *tf-idf* values.

```
  Data: $\mathcal{C}_{u_i}$: the set of closed itemsets of user $u_i$ from Charm;
  $p$ : the number of selected own patterns;
  Result: $\mathcal{P}_{u_i}$: the set of own patterns of user $u_i$;
1 begin
2     Compute the $tf \times idf$ for each pattern from Charm;
3     Sort the list of patterns in descending order according to the $tf \times idf$
      value;
4     Return the top $p$ patterns;
```

3.3 User profiles computation

We define and we denote $\mathcal{P}_{all} = \bigcup_i P_{u_i}$ the whole set of own patterns. The set $\mathcal{P}_{all}$ allows us to define a common space in which all users could be embedded. More formally, $\mathcal{P}_{all}$ defines a vector space $\mathcal{V}$ of size $all = ||\mathcal{P}_{all}||$ where a given user u_i is represented as a vector $V_{u_i} = (m_{i,1}, m_{i,2}, ..., m_{i,all})$.
The second step of our model, called *user profile computation*, is to compute, for each user u_i, a numerical value for each component $m_{i,j}$ of the vector V_{u_i}. i is the user id, $j \in [1, all]$ is a pattern *id* and m stands for a given *measure*. In this paper, we compare two measures proposed in [1]: the *support* and the *lift*.

$$m_{i,j} = support_{\mathcal{S}_{u_i}}(P_j) \qquad and \qquad m_{i,j} = lift_{\mathcal{S}_{u_i}\mathcal{S}}(P_j)$$

3.4 Authentication stage

In our model, the authentication step is based on the identification. For that purpose, our model guesses the user corresponding to an anonymous set of sessions, then it checks if the guessed identity corresponds to the real identity. From this set of sessions we have to build a test profile and to find the nearest user profile defined during the learning step.

Test sessions Performance of our models are calculated on anonymous data sets of growing size.The more information available, the better the classification will be. The first data set consists of only one session, the second consists of 10 sessions, the third one consists of 20 sessions, and the last one consists of 30 sessions. For the test phase, all sessions have the same size of 10 sites.

Building test profile Let $\mathcal{S}$ be the whole set of sessions from the learning data set. Let $\mathcal{S}_{u_t}$ be an anonymous set of sessions and $V_{u_t} = (m_{t,1}, m_{t,2}, ..., m_{t,all})$ its corresponding profile vector. We will compare two approaches to build the anonymous test profile, the *support* and the *lift*:

$$\forall i,\ m_{t,i} = support_{\mathcal{S}_{u_t}}(P_i) \qquad and \qquad \forall i,\ m_{t,i} = lift_{\mathcal{S}_{u_t}\mathcal{S}} = \frac{support_{\mathcal{S}_{u_t}}(P_i)}{support_{\mathcal{S}}(P_i)}$$

Distance functions Let $V_{u_i} = (m_{i,1}, m_{i,2}, ..., m_{i,all})$ and $V_{u_t} = (m_{t,1}, m_{t,2}, ..., m_{t,all})$ be two profiles. We denoted $Dis_{Euclidean}(V_{u_i}, V_{u_t})$ the Euclidean distance and we denote $Sim_{Cosine}(V_{u_i}, V_{u_t})$ the cosine similarity function. We have:

$$Dis_{Euclidean}(V_{u_i}, V_{u_t}) = \sqrt{\sum_j (m_{t,j} - m_{i,j})^2}$$

$$Sim_{Cosine}(V_{u_i}, V_{u_t}) = \frac{\sum_j (m_{t,j} \times m_{i,j})}{\sqrt{\sum_j (m_{t,j})^2 \times \sum_j (m_{i,j})^2}}$$

4 Experimental results

4.1 Data set

Our data set is comprised of the web navigation connection logs of 3,000 users over a six-month period. We have at our disposal the domain name visited and each user ID. From the variables of day and time of connection we have constructed connection sessions for each user. A session is therefore a set of web sites visited. The number of visited web sites per session is limited and equal to 10. For the relevance of our study we used Adblock[6] filters to remove all domains regarded as advertising. The majority of users from this data set are not sufficiently active to be of relevance. Therefore, as in [1], we have limited our study to the 2% of most active users and obtained the significant session sets for 52 users. The 30 users most active (who have a large number of sessions) among those 52 users are used in this paper. Table 2 gives the detailed statistics for this data set.

7698 sessions	**Minimum**	**Maximum**	**Mean**	**Standard deviation**
Size	10	10	10	0
#sessions/users	101	733	257	289

Table 2. Descriptive statistics of the used data set: size of sessions (number of visited web sites) and number of sessions per user, for 30 users.

4.2 Experimental protocol: a description

Algorithm 2 (see appendix) describes our experimental protocol. The first loop sets the size of the set of users among which a group of anonymous sessions will be classified. The second one sets the size of this sessions group. Finally, the third loop sets the number of iterations used to compute the average accuracy rate. The loop on line 10 computes the specific patterns of each user and establishes the profiles vector. The loop on line 13 computes the vector's components for each user. The nested loops on lines 16 and 18 classify test data and compute the accuracy rate.

4.3 Comparative performance of H_1, H_2 and H_3

From own patterns of each user we compute the set $\mathcal{P}_{all}$ as the whole set of own patterns which defines the profile vector of each user. We use the support of a pattern as numerical value for each components (cf. section 3.3). Following Table 3 provides the size of the profile vector and the distribution of own patterns according to size for each heuristic. With 30 users and 10 own patterns per user, the maximal size of the profile is 300.

	Number of own patterns	$\|1\|$	$\|2\|$	$\|3\|$	$\|4\|$	$\|5\|$	$\|6\|$	$\|7\|$
H_1	199	18%	31%	26%	16%	7%	2%	0%
H_2	167	57%	29%	9%	3%	1%	1%	0%
H_3	199	24%	20%	18%	14%	10%	9%	5%

Table 3. Profile vector size and the distribution of own patterns according to size.

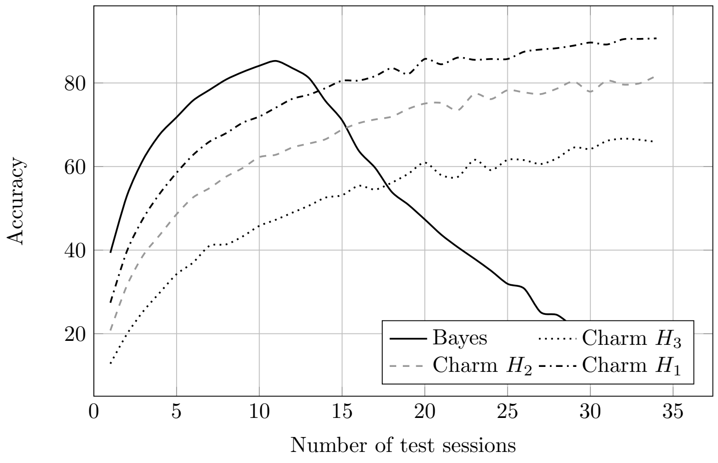

Fig. 2. Comparative performance of H_1, H_2 and H_3. These observations are plotted on an X-Y graph with number of sessions of the anonymous set on the X-axis and accuracy rate on the Y-axis. Measured values are smoothed on 50 executions.

Figure 2 shows that naive Bayes classifier is the most effective if the group of test sessions is from 1 to 13 sessions (10 to 130 visited web sites). This result is in line with the study in [1]. Finally, this graph clearly shows that heuristic H_1 certainly stands out from H_2 and H_3. So, the best heuristic is to choose owns patterns amongst closed patterns with the largest $tf \times idf$ values. As a consequence, the majority of patterns are small-sized patterns (two or three sites) (cf. Table 3). But accuracy rates are much higher.

4.4 Comparative performance with [1]

In [1], the author compares, in particular, two methods of profile vector calculus. In both cases, the own patterns are size 1 and are chosen amongst the most frequent. The first method, named support-based profiling, uses the corresponding support pattern as the numerical value for each component of the profile vector. The second method, called lift-based profiling, uses the lift measure. In order to compare the performances of the H_1 model with the two models support-based profiling and lift-based profiling, we have accurately replicated the experimental protocol described in [1] on our own data set. The results are given in Table 4.

The data of Table 4 highlight that the H_1 heuristic allows rates that are perceptibly better than those of the two models proposed in [1] in all possible scenarios. Nevertheless, the Bayes classifier remains the most efficient when the session group is size 1 in compliance with [1]. Figure 3 allows a clearer understanding of the moment the Bayes curve crosses the H_1 heuristic curve.

[6] http://adblock-listefr.com/

# of users		1	10	20	30
2	Support	65	89	95	97
	Lift	67	90	97	98
	Charm H_1	72	98	**99**	**100**
	Bayes	**85**	**99**	73	61
5	Support	40	74	83	88
	Lift	41	78	86	88
	Charm H_1	49	90	**95**	**98**
	Bayes	**67**	**96**	56	34
10	Support	27	66	79	80
	Lift	29	64	77	80
	Charm H_1	37	83	**92**	**94**
	Bayes	**54**	**91**	51	24
20	Support	19	55	68	75
	Lift	21	58	68	74
	Charm H_1	30	76	**86**	**90**
	Bayes	**43**	**87**	48	19
30	Support	16	53	64	70
	Lift	17	54	64	69
	Charm H_1	26	72	**83**	**89**
	Bayes	**39**	**83**	46	19

Table 4. On left, we find the number of users and the selected model. Each column is defined by the number of sessions of the anonymous data set. Sessions are of size 10. Measured accuracy rate are smoothed on 100 executions. In bold the best values are presented.

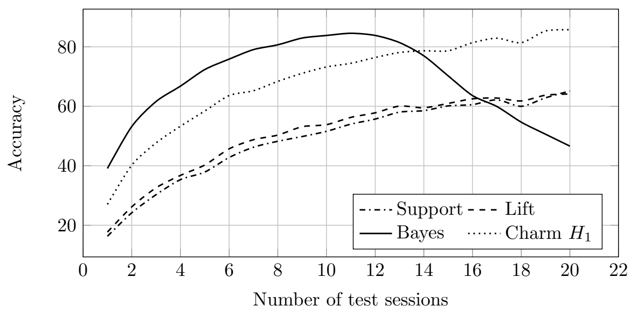

Fig. 3. Comparative performance of Bayes, *support-based profiling*, *lift-based profiling* and H_1. These observations are plotted on an X-Y graph with number of sessions of the anonymous set on the X-axis and accuracy rate on the Y-axis. Number of users is equal to 30. Measured values are smoothed on 50 executions.

4.5 Comparative performance of distance functions

The last figure 4 shows the impact of distance function choice on performances of models.

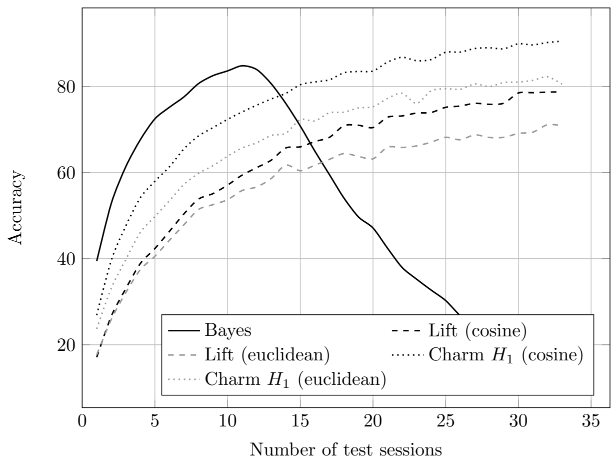

Fig. 4. Comparative performance of both H_1 with cosine similarity and Euclidean distance, Bayes and *lift-based profiling*. These observations are plotted on an X-Y graph with number of sessions of the anonymous set on the X-axis and accuracy rate on the Y-axis. Number of users is equal to 30. Measured values are smoothed on 100 executions.

Figure 4 illustrates the significance of the distance function concerning the performance. Indeed, when used with Euclidean distance, the H_1 method is a bit more precise than the lift one (about 3%). However, performances are improved by using the cosine similarity and their relative ranking is even reversed. H_1 method's performance are then better than lift by 10%.

5 Conclusions and future work

In this study, we proposed a learning model for implicit authentication of web users. We proposed an simple and original algorithm (cf. Algorithm 1) to get a set of own patterns allowing to characterize each web user. The taken patterns have different size and qualify as closed patterns from closure system generated by the set of sessions (cf. Table 3). By reproducing experimental protocol described in [1], we showed that the performances of our model are significantly better than some models proposed in the literature (cf. Table 4). We also showed the key role of the distance function (cf. Figure 4).

This study should be extended in order to improve the obtained results. For a very small sites flow, the results of the solution should be better than results from Bayes' method. Another way to improve results will be to select other types of variable and to add them to our current dataset. The selection of data has an undeniable impact on the results.

References

1. Yang, Y.C.: Web user behavioral profiling for user identification. Decision Support Systems (49) (2010) pp. 261–271
2. Guvence-Rodoper, C.I., Benbasat, I., Cenfetelli, R.T.: Adoption of B2B Exchanges: Effects of IT-Mediated Website Services, Website Functionality, Benefits, and Costs. ICIS 2008 Proceedings (2008)
3. Lagier, F.: Cybercriminalité : 120.000 victimes d'usurpation d'identité chaque année en france. Le populaire du centre (in French) (2013)
4. Filson, D.: The impact of e-commerce strategies on firm value: Lessons from Amazon.com and its early competitors. The Journal of Business **77**(S2) (2004) pp. S135–S154
5. He, R., Yuan, M., Hu, J., Zhang, H., Kan, Z., Ma, J.: A novel service-oriented AAA architecture. **3** (2003) pp. 2833–2837
6. Stockinger, T.: Implicit authentication on mobile devices. The Media Informatics Advanced Seminar on Ubiquitous Computing (2011)
7. Shi, E., Niu, Y., Jakobsson, M., Chow, R.: Implicit authentication through learning user behavior. M. Burmester et al. (Eds.): ISC 2010, LNCS 6531 (2011) pp. 99–113
8. Jakobsson, M., Shi, E., Golle, P., Chow, R.: Implicit authentication for mobile devices. Proceeding HotSec'09 Proceedings of the 4th USENIX conference on Hot topics in security (2009) pp. 9–9
9. Ullah, I., Bonnet, G., Doyen, G., Gaïti, D.: Un classifieur du comportement des utilisateurs dans les applications pair-à-pair de streaming vidéo. CFIP 2011 - Colloque Francophone sur l'Ingénierie des Protocoles (in French) (2011)
10. Goel, S., Hofman, J.M., Sirer, M.I.: Who does what on the web: A large-scale study of browsing behavior. In: ICWSM. (2012)
11. Kumar, R., Tomkins, A.: A characterization of online browsing behavior. In: Proceedings of the 19th international conference on World wide web, ACM (2010) pp. 561–570
12. Abramson, M., Aha, D.W.: User authentication from web browsing behavior. Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference pp. 268–273
13. Herrmann, D., Banse, C., Federrath, H.: Behavior-based tracking: Exploiting characteristic patterns in DNS traffic. Computer & Security (2013) pp. 1–17
14. Salton, G.: Automatic text processing: The transformation, analysis and retrieval of information by computer. Addison Wesley (1989)
15. Davey, B.A., Priestley, H.A.: Introduction to lattices and orders. Cambridge University Press (1991)
16. Ganter, B., Wille, R.: Formal concept analysis, mathematical foundation, Berlin-Heidelberg-NewYork et al.:Springer (1999)
17. Zaki, M.J., Hsiao, C.: Efficient algorithms for mining closed itemsets and their lattice structure. IEEE Transactions on knowledge and data engineering **17**(4) (2002) pp. 462–478
18. Szathmary, L.: Symbolic Data Mining Methods with the Coron Platform. PhD Thesis in Computer Science, University Henri Poincaré – Nancy 1, France (Nov 2006)

Appendix

Algorithm 2: Experiment procedure

```
Data: $\bigcup_i \mathcal{S}_{u_i}$ : all sessions from $n$ users;
$X$ : number of successive executions;
Result: The mean accuracy of select models;
1  begin
2      for ($N = \{2, 5, 10, 20, 30\}$) do
3          for ($S = \{1, 10, 20, 30\}$) do
4              for ($z = 1, \ldots, X$) do
5                  Select $N$ random users;
6                  For each user, select $SN = min(|\mathcal{S}_{u_i}|,\ i = 1, \ldots, N)$;
7                  Take the $\frac{2}{3}$ of the $SN$ sessions from each users to form the
                   training set;
8                  Take the rest of $SN$ sessions to form the validation set;
9                  $\mathcal{P}^k_{all} \leftarrow \emptyset$ (the global profile vector for each model $k$);
10                 for each ($u_i,\ i = 1, \ldots, N$) do
11                     Compute the own patterns $\mathcal{P}^k_{u_i}$ $(1 \leq |\mathcal{P}^k_{u_i}| \leq 10)$;
12                     $\mathcal{P}^k_{all} \leftarrow \mathcal{P}^k_{all} \cup \mathcal{P}^k_{u_i}$;
13                 for each ($u_i,\ i = 1, \ldots, N$) do
14                     Compute the vector $V^k_{u_i}$ with support or lift;
15                 Initialize to 0 the confusion matrix $M^k$ of the method $k$;
16                 for each ($u_i,\ i = 1, \ldots, N$ do
17                     Compute the test stream $\mathcal{T}_{u_i}$ ($|T|$ is fixed, $T \in \mathcal{T}_{u_i}$);
18                     while ($\mathcal{T}_{u_i} \neq \emptyset$) do
19                         Take $SW$ sessions from $\mathcal{T}_{u_i}$ to compute $V^k_T$;
20                         $u_a \leftarrow max(simil(V^k_{u_i}, V^k_T))$ or $min(dist(V^k_{u_i}, V^k_T))$;
21                         $M^k[u_i][u_a] \leftarrow M^k[u_i][u_a] + 1$;
22             Compute the mean accuracy of $k$ from $M^k$;
```