Towards a Personal Identity Code Respecting Privacy

Denis Migdal, Christophe Rosenberger

Towards a Personal Identity Code Respecting Privacy

Denis Migdal, Christophe Rosenberger
Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France
denis.migdal, christophe.rosenberger}@ensicaen.fr

February 16, 2018

Abstract

Various applications on Internet require information on users, to verify their right to access services (verification of identity proofs s.a. passwords), to avoid attacks (s.a. paedophilia, profile usurpations), or to give trust to users (e.g. in social networks). In this paper, we introduce a method to generate (non-cryptographics) identity-based signatures computed from 1) collection of data from user biometrics, computer configuration, web browser fingerprinting, 2) data pre-processing, 3) protection of personal information through generation of a binary code (our signature). We illustrate the benefits of the proposed method with preliminary results on real personal information.

1 Introduction

Consumption of Internet services is nowadays significant, whether it concerns social networks, e-commerce, or online games. E.g. in 2016, 96% of French criminal records queries have been done through Internet (Sta,). However, several personal information can be collected during usage of Internet services, either given by users (s.a. on social networks), or automatically collected. Internet services collect more and more personal information linked to users, sometimes for legitimate usage (s.a. fraud detection, remote examinations), but also for non-compliant uses with collection terms (s.a. sales to other services, identity consolidation). Such information might be linked to the user (s.a. biometrics data, name, age), to the browser (s.a. version, type), to the device (s.a. operating system, hardware, screen resolution). The collection of such information, even in a legitimate context, might enable user identification, posing major privacy issues.

This paper main contribution is the proposal of a method to generate binary codes linked to users identities. This code does not enable to retrieve information used for its computation, but efficient comparison with other codes is possible through their Hamming distances. We introduce the several computations steps of this code. Used informations go, from the browser, to the user's device. Collected information is pre-processed, enabling the computation of the code in the last step. Usage of such codes are not in the scope of this article, however we briefly introduce few interesting applications (authentication, attacks identification s.a. multi-accounts detection). This paper is organized as follows. Section 2 introduces previous works on personal information collection and usage. The proposed method is described in section 3. Section 4 introduces preliminary results on real information. We conclude and give some perspectives in Section 5.

2 Previous works

Browser Fingerprinting allows tracking user's browser thanks to discriminant data a service can collect. This is usually proposed to "personalize services" corresponding to users profile-type. Panopticlick (Eckersley, 2010), IAmUnique (Laperdrix et al., 2016), and UniqueMachine (Cao and Wijmans, 2017) websites enable computation of browser fingerprints from data collected by the website, generally through the network and JavaScript API, to determine the fingerprint uniqueness among the previous computed. The more the browser fingerprint is unique, the more the service is able to discriminate the user. However, browser fingerprint might vary, e.g. by changing of browser, its configuration (Nikiforakis et al., 2015), or device. The goal is not to identify users with assurance, but to identify a set of browsing session belonging to the same user, or type of users.

Information used for browser fingerprinting might be linked, e.g. to the hardware (e.g. GPU (Cao and Wijmans, 2017), screen), to the operating system, to the browser, its configuration, installed fonts (Eckersley, 2010; Laperdrix et al., 2016), browser history (Weinberg et al., 2011), or blacklisted domains (Boda et al., 2012).

Keystroke dynamics is a behavioral biometric modality consisting in analyzing users' way of typing on a keyboard. This biometric information can be computed easily on Internet using a simple JavaScript code. Keystroke dynamics has been experimented for the first time in 1980 in a study where seven secretaries were asked to type three different texts (Gaines et al., 1980). The results were promising, but lacked a sufficient number of users involved in the database. The first patent on keystroke dynamics was registered in 1986 (Garcia, 1986). Other methods have been defined during the last twenty years (Phoaha et al., 2009). In previous references such as (Giot et al., 2009), it has been shown that keystroke dynamics is

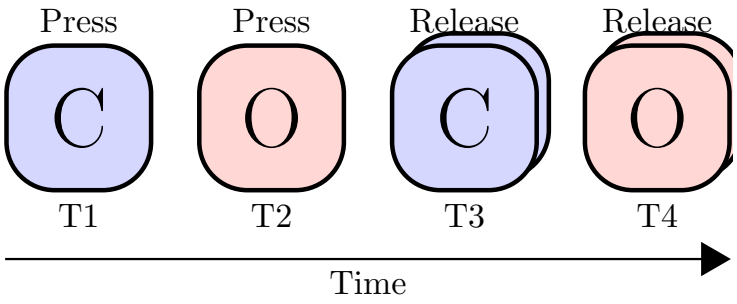


Figure 1: Information captured in a keystroke dynamics system when pressing C and O keys (Giot et al., 2011).

invariant to the keyboard type (laptop or terminal). The use of mobile devices is not considered in this paper but many methods exist to deal with this type of capture (Dafer and El-Abed, 2017). The capture process of keystroke dynamics is presented in Figure 1. It consists in computing several features when the keys are pressed and released (timestamp of the event, code of the key, ...) provided by any Operating System (OS). The feature extraction consists mainly in computing different latencies and duration times between each key. Figure 1 shows an example where the user presses two keys of the keyboard. The user presses "C" at T1, "O" at T2 and releases "C" at T3 and "O" at T4. Note that the following relation is always respected: $T3 > T1$ and $T4 > T2$ (we always release a key after pressing it), while the following condition may not always be respected: $T2 > T3$ (because, as in our example, a user may press another key before releasing the previous one). We can extract three different types of latencies (T2-T1, T4-T3,T2-T3) which we call PP (latency between two pressures), RR (latency between two releases), RP (latency between one release and one pressure) respectively and one type of duration (T3-T1 or T4-T2) which we call PR (duration of a key press). The described process is repeated for all the keys.

All these previous works are able to collect a lot of personal data related to one specific user. The problem is that they are not protected, or only with a simple hashing function (that does not permit to compute similarities between different signatures). We intend to solve this problem.

3 Proposed method

The goal of the proposed method is to compute a binary code linked to an user from personal information (technical and biometrics). This code must answer several requirements:

- *Non reversibility*: the user binary code must not give information about the collected personal information.
- *Confidentiality*: the attribute value cannot be known, nor deducted, by the service.
- *Similarity conservation*: If users' personal information are similar, then their binary code must be too (Hamming distance).
- *Non-usurpation*: a tiers cannot forge a code enabling him/her to usurp legitimate users' identity.
- *Revocation*: legitimate user must be able to revoke an existing binary code.

Figure 2: Proposed method principle

In the scope of article, a trust score can be computed with the Hamming distance between the proof and the commitment, both fixed-size binary vectors. Therefore, we consider and detail the following personal information modality:

- what the user is/knows to do: its behavioural biometric;
- what the user has: its browser;
- where the user is: its physical and organizational localization;
- "what the user prefers": personal machine configuration.

Figure 2 introduces the general principal of the proposed method. A password is used as a secret key (Lacharme and Plateaux, 2011). In this case, Alice by inputting her password consent to give the binary code to the service. The different computation steps are introduced later.

3.1 Collection of personal information

Nowadays, it is possible to collect a large number of personal information. We detail the collected information by categories.

3.1.1 Browser

To authenticate the browser, a simple key, stored on it is enough. The key, we named *localkey*, is an n-bits value randomly generated upon first usage of the browser. This key is then used to authenticate the browser. For n big enough, the probability of collisions is insignificant, and the exhaustive research, hard. In the frame of the experience, n=64, for higher security needs, the key size may be increase, e.g. with n=512. The key might be stored in the browser localStorage[1], or, ideally, in a WebExtension simple-storage. Nonetheless, it is possible for an attacker to steal the key if he has access to the device, or to the user session. The keys being randomly generated, the theft of one do not compromise the others possessed by the user. The key might be protected, e.g. with encryption, or the fraudulent usage be detected, e.g. with others personal information. However, this will not be introduced in the frame of this article.

[1]HTML5 feature

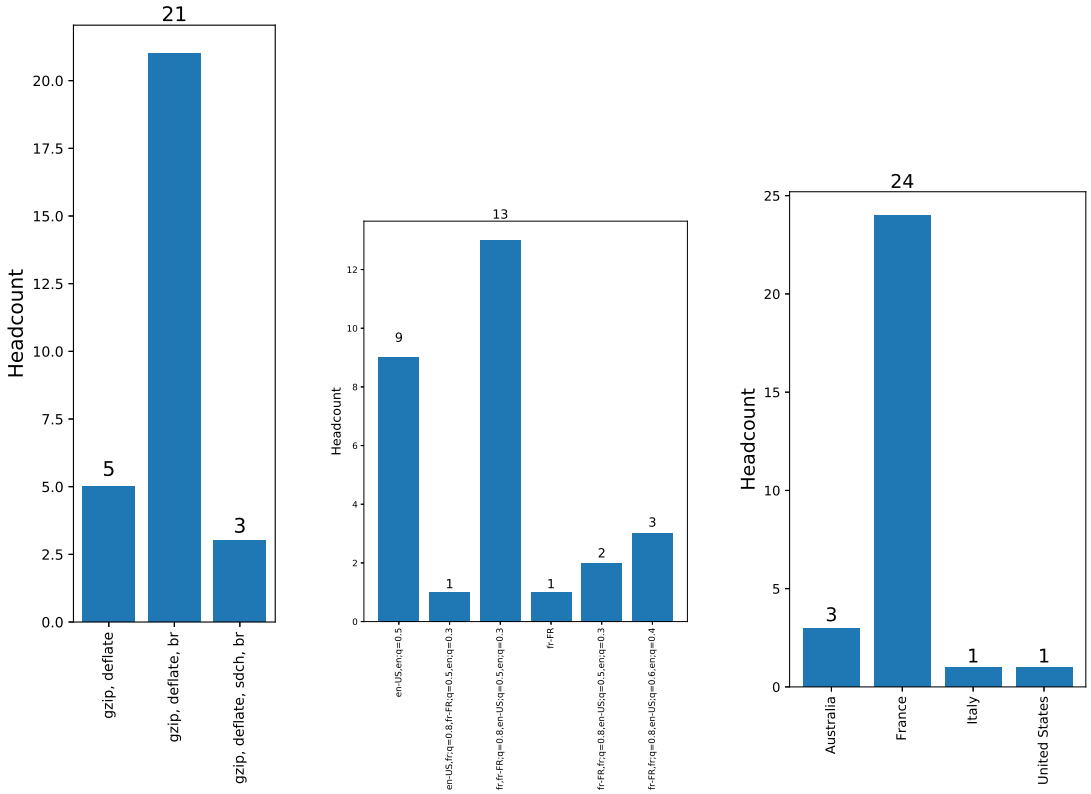


Figure 3: Personal information collection: distribution of some collected data (raw data)

3.1.2 Localization

IP addresses are distributed by ranges, from IANA[2] to RIR[3], from RIR to LIR[4], and finally from LIR to users. It is then possible to deduce from it the user network, the administrative (e.g. county) or physical (e.g. GPS position) localization. However, the TOR network, a VPN, or a proxy might be used to masquerade the user IP address. Then, the network and locations deduced from the IP address would be the proxy, VPN, or exit TOR nodes. In the scope of this article, the administrative (country, region, county, town) and physical (latitude and longitude) are extracted through the Google MAP API from an address extracted from the database dp-ip[5]. In a future work, it would be possible to deduce either the user's ISP (Internet Service Provider), or the structural localization among an entity (e.g. company, university, research center, state agencies), thanks to DNS, reverse DNS, WHOIS IP, and WHOIS domain queries. It is also possible to get more information about the IP address thanks to DNSBL[6].

[2]Internet Assigned Numbers Authority
[3]Regional Internet Registry
[4]Local Internet Registry
[5]download.dp-ip.com/free/dbip-city-2017-05.csv.gz
[6]DNS Blacklist

3.1.3 Network data

Data sent to the service by the communication protocols are discriminant and enable, by browser fingerprinting techniques, user identification (Eckersley, 2010; Laperdrix et al., 2016). In the same way, such data can be used for user authentication by comparing them to enrolment data. As a consequence, this modality cannot be used if the data are randomized for each transaction. However, usurpation is trivial for whom knows this data, e.g. for whom provide a service to the user. Moreover, the usage of normalized data, e.g. by user the TOR browser, increases the collision probability. This modality gives little trust in the user authentication, but enable to detect reception of unusual data. In the scope of this article, the following fields are extracted from the HTTP header:

- *User-Agent*: arbitrary string defined by the browser;
- *Accept, Accept-Language, Accept-Encoding*: formats, languages, and encoding preferences (values $\in [0, 1]$);
- *Referer*: previous pages URL, sometimes randomized, truncated, or removed;
- *Cookie*: cookies sent by the browser;
- *DNT, Connection, Upgrade-Insecure-Requests*: other parameters.

Figure 3 shows the distribution of some network and browser data for different users. We can see clearly some differences for each user even if most of them are french. Figure 4 shows the distribution of the User-agent value that are very discriminant among users.

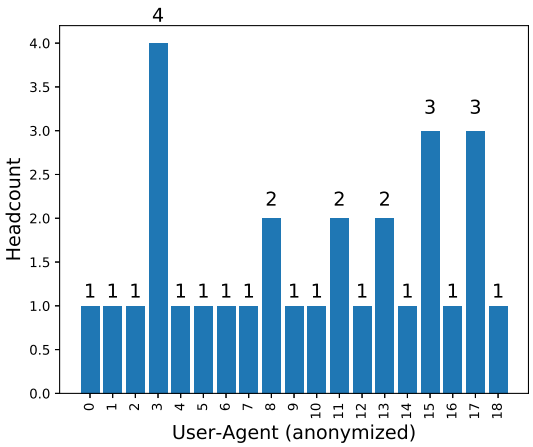


Figure 4: Personal information collection: distribution of the user agent (raw data)

3.1.4 Biometric data

User behavioural biometric can be analysed from keyboard and mouse actions, captured by JavaScript events in the browser. In the scope of this article, the user keystroke is computed from the 20 most frequent digrammes: *"r ", "te", "nt", ", ", "n ", "en", " s", "le", " l", " c", "de", ('arrowleft', 'arrowright'), " p", " d", "on", "t ", "es", "s ", "e ", ('backspace', 'backspace')*. The following durations are measured:

- P_1R_1: first character pressure;
- P_2R_2: second character pressure;
- P_1P_2: between the two characters push;
- R_1R_2: between the two characters release;
- R_1P_2: between the first character release and the second push;
- P_1R_2: between the first character push and the second release.

3.2 Data pre-processing

To obtain, for each modality, a fixed-size real vector (required for the protection scheme), collected data are converted to real vectors then appended. The distance between two vectors might be influenced by extremes values, they are consequently normalized.

3.2.1 Browser

Localkey (n-bits key) is converted into a n-bits vector. Thus, the 16-bits localkey *"0x0123"*, is converted into *[0,0,0,0, 1,0,0,0, 0,1,0,0, 1,1,0,0]*.

3.2.2 Localisation

An IP address is converted in a vector composed by:

- a vector composed by the IP address bits divided by 2^{32-p-1} with p (bit weight);
- a vector composed by the $128/2^k$ first bits of the locality name's md5 hash with k=1 for "country", k=2 for "region", k=3 for "county", and k=4 for "town";
- a vector composed of 3 angles $\in [-90;+90]$ representing the GPS localization's latitude (lat), and the longitude l (lng1, lng2); lng1 and lng2 are equal to:

  $$sign(\alpha) * ||\alpha| - (|\alpha| > 90) * 180|$$

  with $\alpha = l$ for lng1 and $\alpha = rot90(l) = (l-90)\%360 - 180$ for lng2. These angles in degree are normalized by the following formula:

  $$angle^* = (angle + 90)/180$$

As for example, the IP adress *"127.0.0.1"* is converted in *[0, 0.5, 0.25, 0.125, 0.0625, 0.03125, 0.015625, 0, 0.0078125, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,* $4.6566 * 10^{-10}$*]*. The following GPS localization *(135, 0)* is converted in *[0.5, 0.75, 0.25]*.

3.2.3 Network data

Referer, User-Agent, Connection and *Cookie* are converted into histograms, vectors giving for each character its headcount. Only the ASCII characters $\in [0x20, 0x7F[$, so 95 characters, are considered. *Accept, Accept-Encoding,* and *Accept-Language* are converted into vectors giving the preference for each format, encoding, and language from a predefined list. An additional value indicates the presence of spaces after comma in the field. *DNT* and *Upgrade-Unsecure-Requests* are converted into a 1-integer vector, equals to 1 if setted, 0 otherwise. The predefined lists are:

- Accept: "text/html", "application/xhtml+xml", "application/xml", "image/webp", "image/jxr";

- Accept-Encoding: ”gzip”, ”deflate”, ”br”, ”sdch”;
- Accept-Language: ”fr”, ”fr-FR”, ”en-US”, ”en”.

As for exemple, the following User-Agent value *”Browser/1.0 (Operating System; rv:1.0) Engine/20170701 Browser/1.0”* is converted by considering only characters in [a-z] by *[1, 0, 0, 0, 5, 0, 2, 0, 2, 0, 0, 0, 1, 3, 2, 1, 0, 6, 3, 2, 0, 1, 2, 0, 1, 0]*. The Accept-Language *”fr;q=0.8, fr-FR;q=0.5, en-US”* is described by *[0.8, 0.5, 1, 0, 1]*. The DNT value *”1”* is converted in *[1]*.

3.2.4 Biometric data

The collected durations are converted into a vector giving, for each considered digram, the means of the 6 durations. These average values are converted in milliseconds, limited by 1000 then divided by 1000. Figure 5 presents the signature values after pre-processing (here 1218 values). This step permits to protect the semantic content of the signature, we propose to enhance this protection thanks a dedicated process presented in the next section.

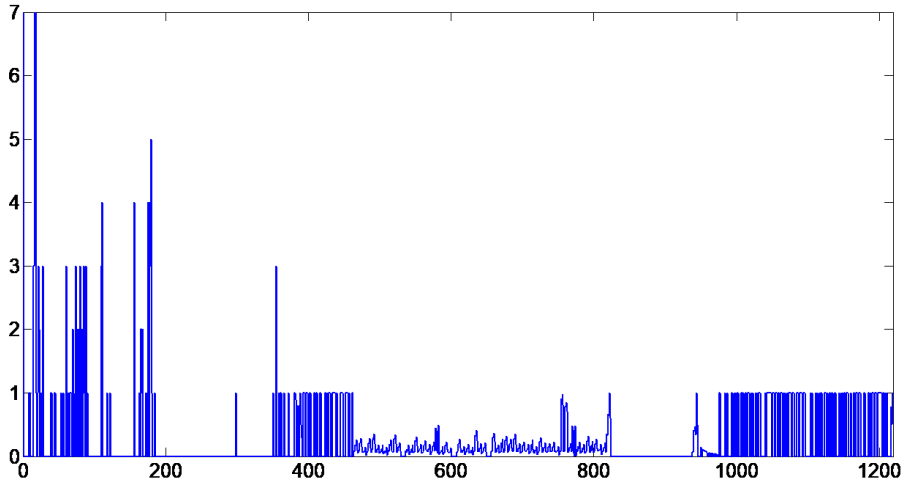


Figure 5: Example of raw values after pre-processing (1218 real values)

3.3 Data protection

The issue we want to address in this work is the possibility to answer to Internet services applications (s.a. authentication, attacks detection) while preserving the user privacy. From the personal information collected, we aim at generating a binary signature as dynamical user characteristics having lost its semantic description. Finally, the service is able to exploit this signature without knowing the information used to generate it.
Biohashing is a well-known algorithm in biometrics. It enables a biometric data transformation when represented by a fixed-size real vector. It allows the generation of a binary model called BioCode having a size inferior or equal to the original size. This transformation is non-reversible and allows to keep input data similarity. This algorithm originally has been proposed for face and fingerprints by

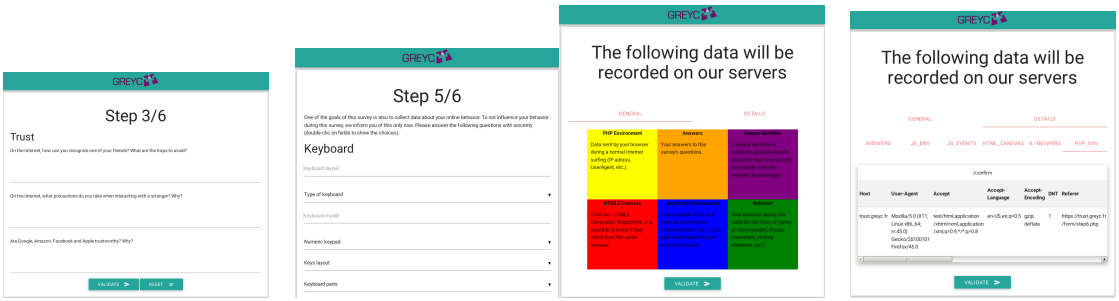


Figure 6: Personal information collection questionnaire's screens

Teoh *et al.* in (Teoh et al., 2004). Biohashing algorithm can be used on every biometric modality, or personal information, that can be represented by a fixed-size real vector. This transformation requires a secret linked to the user. In our case, this could be a password input by the user (Lacharme and Plateaux, 2011). The BioCode comparison is realized by the computation of the Hamming distance. The BioHashing algorithm transforms a parameter vector $T = (T_1, \dots T_n)$ into a binary model called BioCode $B = (B_1, \dots B_m)$, with $m \leq n$, as follows:

1. m random orthonormal vectors $V_1, \dots, V_m$ of length n are generated from a secret used as a seed for random draw (typically with the Gram Schmidt algorithm).
2. For $i = 1, \dots, m$, compute the dot product $x_i =< T, V_i >$.
3. BioCode computation $B = (B_1, \dots, B_m)$ with the quantization process:

$$B_i = \begin{cases} 0 & \text{if} \quad x_i < \tau \\ 1 & \text{if} \quad x_i \geq \tau, \end{cases}$$

   Where τ is a given threshold, generally equals to 0.

The algorithm performance is granted by the dot product with orthonormal vectors, as detailed in (Teoh et al., 2008). The quantization process guarantees the data non-reversibility (even if $n = m$), as each input coordinate T is a real value, when the BioCode B is binary. We propose the use of this transformation to protect personal information.

4 Experiments

In this section, we first detail the used experimental protocol. Second, some preliminaries results are given to show the binary code computation benefits.

4.1 Experimental protocol

An acquisition campaign as be organized in march 2017 in the trust.greyc.fr website. The participants have been recruited from the GREYC laboratory and the engineering school ENSICAN, broadcast lists. Thus, collected data come from an unique place, indeed the majority of the participants are localized in Caen, use the same networks (ENSICAN and UNICAEN), and thus have the same IP address.

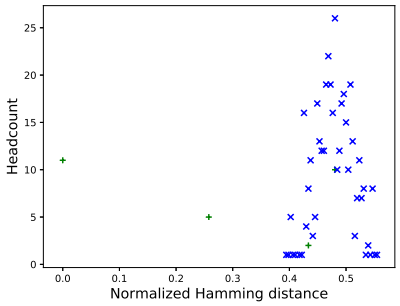
Moreover, the use of GREYC and ENSICAEN devices make the participants configuration, and network data quite similar. With only 22 participants, mostly located in Caen, the sample is not representative, but enables a first experimentation of the personal identity code. During the acquisition, participants are invited to answer to 8 questions on privacy, then to copy an extract of the Universal Declaration of Human Rights (see 6). To prevent any influence for the keystroke dynamics, participants are informed of the information collection only from the step 5, where they are invited to give the authorization to use personal information for research purposes. All the collected data are stored in the browser sessionStorage and are submitted only after user validation through the confirmation page, resuming the collected information types, and detailing collected information. Once the data submitted, a localkey is generated and stored inside the browser localStorage, to recognize the browser upon multiples submissions. The localkey is also printed to users so that they can exercise their right of data access and correction.

4.2 Experimental results

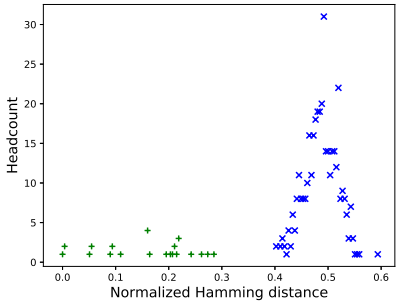
From the 29 collections, from 22 users (8 have been made by the same user in different contexts), we estimate in which proportion these information enable to compute users similarity. Figure 7 presents the distribution of BioCodes comparisons for all users using different collected information and the total. In green, are represented intra-users comparisons between BioCodes and in blue inter-users comparisons. The distribution of BioCodes generated by taking into account only localization (Figure 7 (a)) show some errors to discriminate users. Indeed, the same user provided some information at different localizations (sometimes more similar to other users). The BioCode generated using the PHP environment and the total, permits to clearly discriminate users from each others.

Figure 8 shows two distance matrices. The first (a) compare pre-processed data (without any protection) with the cosine distance ($1 - cos(A,B)$, if A and B are two real vectors. In this figure, we can notice two things. The first is that the signatures 4 and 5 are judged very similar. This is in fact the same user in the same context. The only difference is in the keystroke dynamics. Signatures 3 to 10 have been generated by the same user, but in different contexts (s.a. Wifi, browser), the similarity is more contrasted. The second important observation is the relative similarity of the signatures 4 and 5 with others signatures. This can be explained as these signatures have been acquired inside the laboratory with devices with similar configurations and IP address. Figure 8 (b) represents the distances between BioCodes (protected signatures) with, for each user, an unique secret key. With the protection and these keys, we highlight the similarity between users. For binary codes linked to signatures 3 to 10, we identify a similarity between then with variations depending on the similarity of personal information. This demonstrates the capacity of the proposed method to produce an exploitable code for personal information similarity computation.

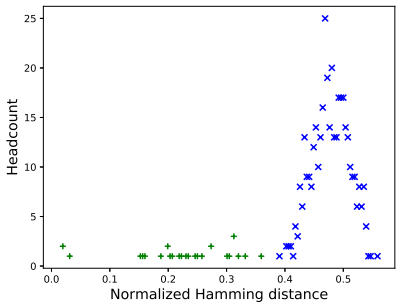
Regarding the requirements previously announced, it is easy to verify their accordance. The BioHashing algorithm used to compute the binary BioCode allows non-reversibility and the capability to compute the similarity of users identity. The data confidentiality is obtained with this last transformation and a secret key usage (here a password). An impostor would not be able to generate such binary code without knowing the secret key, and using the same device. An impostor, at better, replays an existing data. However, the protection of the communication channel, and of the data on the service-side solves this problem. The code revocation is trivial by changing the secret key (i.e. here, the password).



(a)



(b)



(c)

Figure 7: Distribution of BioCode comparisons for all users: (a) localisation BioCode, (b) PhP environment BioCode, (c) total BioCode. In green, are represented intra-user comparisons and in blue inter-users comparisons.

5 Conclusion and perspectives

In this paper, we propose a method enabling to compute a personal code linked to an user while respecting their privacy. This code incorporates different information linked to the browser, keystroke dynamics, or localization. We showed on a preliminary dataset composed of 29 collections that it was possible to obtain a binary code similar for a same user in spite of differences of contexts. Many applications are conceivable from this work s.a. user authentication, multi-account

identifications. These applications constitute this study perspectives.

ACKNOWLEDGMENTS

Authors would like to thank the Normandy Region for the financial support of this work.

REFERENCES

Tableau de bord des services publics numriques dition 2017. http://www.modernisation.gouv.fr/ladministration-change-avec-le-numerique/par-des-services-numeriques-aux-usagers/tableau-de-bord-des-services-publics-numeriques-edition-2017.

Boda, K., Földes, Á., Gulyás, G., and Imre, S. (2012). User tracking on the web via cross-browser fingerprinting. *Information Security Technology for Applications*, pages 31–46.

Cao, S. Y. and Wijmans, E. (2017). Browser fingerprinting via os and hardware level features. *Network & Distributed System Security Symposium, NDSS*, 17.

Dafer, M. and El-Abed, M. (2017). Evaluation of keystroke dynamics authentication systems: Analysis of physical and touch screen keyboards. In *Developing Next-Generation Countermeasures for Homeland Security Threat Prevention*, pages 306–329. IGI Global.

Eckersley, P. (2010). How unique is your web browser? *Privacy Enhancing Technologies*, 6205:1–18.

Gaines, R., Lisowski, W., Press, S., and Shapiro, N. (1980). Authentication by keystroke timing: some preliminary results. Technical report, Rand Corporation.

Garcia, J. D. (1986). Personal identification apparatus. US Patent 4,621,334.

Giot, R., El-Abed, M., Hemery, B., and Rosenberger, C. (2011). Unconstrained keystroke dynamics authentication with shared secret. *Computers & Security*, 30(6):427–445.

Giot, R., El-Abed, M., and Rosenberger, C. (2009). Greyc keystroke: a benchmark for keystroke dynamics biometric systems. In *Biometrics: Theory, Applications, and Systems, 2009. BTAS'09. IEEE 3rd International Conference on*, pages 1–6. IEEE.

Lacharme, P. and Plateaux, A. (2011). Pin-based cancelable biometrics. *International Journal of Automated Identification Technology (IJAIT)*, 3(2):75–79.

Laperdrix, P., Rudametkin, W., and Baudry, B. (2016). Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints. *Security and Privacy (SP)*, pages 878–894.

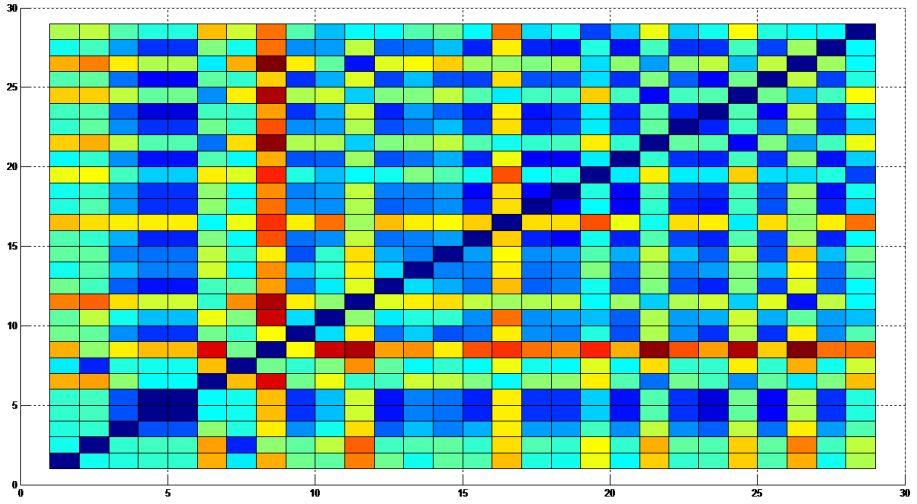
Nikiforakis, N., Joosen, W., and Livshits, B. (2015). Privaricator: Deceiving fingerprinters with little white lies. *Proceedings of the 24th International Conference on World Wide Web*, pages 820–830.

Phoaha, V. V., Phoha, S., Ray, A., Joshi, S. S., and Vuyyuru, S. K. (2009). Hidden markov model (hmm)-based user authentication using keystroke dynamics. patent.

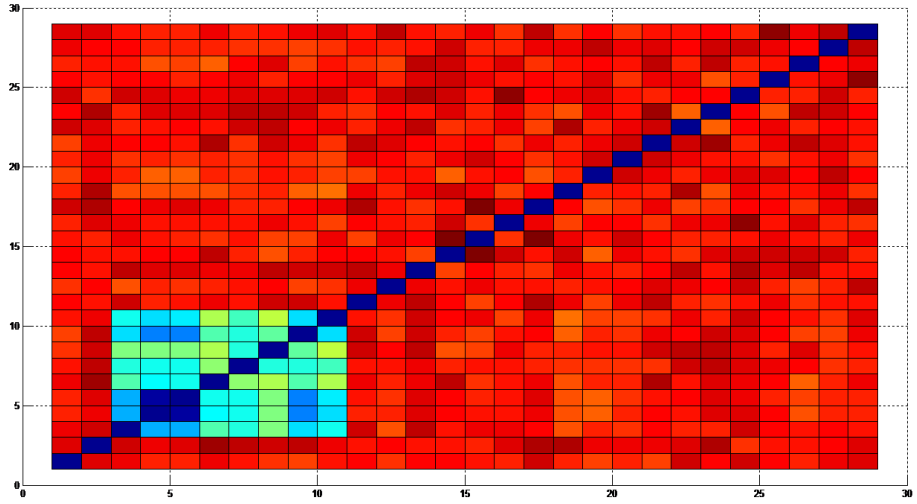
Teoh, A., Ngo, D., and Goh, A. (2004). Biohashing: two factor authentication featuring fingerprint data and tokenised random number. *Pattern recognition*, 40.

Teoh, A. B., Kuan, Y. W., and Lee, S. (2008). Cancellable biometrics and annotations on biohash. *Pattern Recognition*, 41:2034–2044.

Weinberg, Z., Chen, E. Y., Jayaraman, P. R., and Jackson, C. (2011). I still know what you visited last summer: Leaking browsing history via user interaction and side channel attacks. *Security and Privacy (SP)*.



(a)



(b)

Figure 8: Information comparison between the pre-processed data (a) and after protection (b). In coordinates, the compared entries' number. Blue for an high similarity, red for low.