

# Empreinte Sociométrique

CYRIL DEVER

Edgewhere

November 14, 2016

## Abstract

*We describe an algorithm that takes contact data to represent the social footprint of a person in a secure and universal way. It allows recognizing an identity without sharing or exposing any personal information to any potential eavesdropper. It could evolve with time and may carry the whole contact history of a person, making it possible to verify if he or she matches with even very old data, all without needing any kind of disclosure between parties. Last update: this technique is pending patent\*.*

## I. INTRODUCTION

**L**ike a tyre or a shoe leaving a distinctive mark on the ground, or a finger on a glass, each one of us leaves a trace the more specific the richer our social interactions are. In particular, the history of our contact data is every day more distinct to someone else's. Even before we move from our parents house, we start leaving personal trails (a first cell phone, a pseudo we use for a game, ...) and, of course, our full civil status (names, date of birth, etc.).

Of course, we wouldn't want to share all these information to everyone. So, ensuring maximum security is obviously mandatory when it comes to manipulating personal data.

We herein describe a way to build such a safe footprint that we call *Empreinte Sociométrique*[1] and that is both totally secure, thanks to the use of strong pseudonymization techniques, and particularly effective, that is it ensures that, even though two different *Empreintes Sociométriques* don't look quite alike, they might be representing the same person, but no external stakeholder may ever know.

Embedded in a QR Code or used as is, it could become an assistant to any identification device or to further data manipulation, such as secure deduplication and anonymous enrichment, or even blind comparison of customer databases, all in full compliance with the latest regulations (GDPR, CCPA, ...).

## II. FORMAL DESCRIPTION

### 1. GENERAL ALGORITHM

An *Empreinte Sociométrique*<sup>1</sup> takes an input data characterized by its source and its type and operates several operations to form the final elements: its corpus and its signature concatenated into one long string.

**Definition 1** (Source Data). A source data  $\varsigma$  is the actual contact data we want to print in the *Empreinte Sociométrique*, eg. "Cyril".

It is defined in the *words* space:  $\omega$ .

**Definition 2** (Data Type). We define the data type  $\tau$  as a code defining which kind of source data we are dealing with, eg. "firstname".

It is defined in a set of data types  $\mathcal{T}$ <sup>2</sup>.

**Definition 3** (Input Data). We define an input data  $d_i$  as a tuple of data type and source data:

$$d_i := [d_i^\tau, d_i^\varsigma] \text{ with } \begin{cases} d_i^\tau \in \mathcal{T}, \text{ the data type} \\ d_i^\varsigma \in \omega, \text{ the source data} \end{cases} \quad (1)$$

**Definition 4** (Recombined Contact). A recombined contact is a final contact data potentially made out of different input data.

For example, you can create a recombined address4 by concatenating a `streetName` with a `streetNumber`.

Let  $v : \omega \rightarrow \omega$  be a normalization function that takes an input data and returns its normalized counterpart.

\*filed under registration number FR1905778 at INPI on May 29, 2019

<sup>1</sup>a literate translation being a *Sociometric Imprint*

<sup>2</sup>see Table 1 for available values in  $\mathcal{T}$

And let  $\rho : \omega^n \rightarrow \omega$  be a recombination function that takes several input data to build a missing recombined contact.

Finally, let  $h()$  be a cryptographic hashing function<sup>3</sup>,  $c(msg, key)$  an encryption function and  $\zeta()$  a compression algorithm.

---

**Algorithm 1:** Overall construction
 

---

**Input:** A vector  $\mathbf{d} := \{d_1, d_2, \dots, d_n\}$  of input data, a key  $K$   
**Output:** The *Empreinte Sociométrique* or an error

```

1 if  $d = \emptyset$  then
2   throw empty input data set
3 initialize the set of normalized data
    $\mathcal{D} \leftarrow \emptyset$ ;
4 for  $i \leftarrow 0$  to  $n$  by 1 do
5   if  $d_i^\tau \notin \mathcal{T}$  then
6     continue;
7   normalize input data:  $d_{Norm} \leftarrow v(d_i)$ ;
8   if  $d_{Norm} \neq \emptyset$  then
9      $\mathcal{D} \leftarrow d_{Norm}$ ;
10 create the set of recombined contacts  $\mathcal{R}$ 
    from the normalized data:  $\mathcal{R} \leftarrow \rho(\mathcal{D})$ ;
11 initialize the vector of ciphered contacts
     $\mathcal{C} \leftarrow \emptyset$ ;
12 for  $i \leftarrow 0$  to  $\|\mathcal{R}\|$  by 1 do
13    $\mathcal{C} \leftarrow h(\mathcal{R}_i)$ ;
14 initialize the sets of categorized contacts:
     $\mathcal{V}$  the variants, and  $\bar{\mathcal{V}}$  the invariants;
15 for  $i \leftarrow 0$  to  $\|\mathcal{C}\|$  by 1 do
16   if  $\mathcal{C}_i$  is invariant then
17      $\bar{\mathcal{V}} \leftarrow \mathcal{C}_i$ ;
18   else
19      $\mathcal{V} \leftarrow \mathcal{C}_i$ ;
20 build the corpus  $c$  using  $\mathcal{V}$  and  $\bar{\mathcal{V}}$ ;
21 encrypt it and compress it to build the
    Empreinte Sociométrique:
     $ES \leftarrow \zeta \circ c(c, K)$ ;
22 return  $ES$ ;
```

---

Algorithm 1 describes the general steps to take that leads from a set of input data to its

<sup>3</sup>set as a system parameter

*Empreinte Sociométrique*.

In a nutshell, the whole process could be summed up as follows:

- Data handling;
- Normalization of contact data;
- Recombination to maximize endpoints;
- Ciphering of each endpoint;
- Organization of the corpus;
- Encryption;
- Compression;
- Signature and final packaging.

To ensure maximum security, the whole operation should take place on the data owner side.

## 2. STEP-BY-STEP PROCESS

As seen in Algorithm 1, the very first step applied to an input data  $d_i$  is to check whether it is a valid or not.

At this stage, an input data would be deemed invalid for two reasons:

- The source data is empty:  $d_i^s = \emptyset$ ;
- The data type doesn't exist:  $d_i^\tau \notin \mathcal{T}$ .

Each invalid input data is discarded while each valid one is then applied the normalization function  $v()$ .

### 2.1 Normalization

The goal of such a normalization is to homogenize source data to make sure the subsequent operations handle equivalent data.

**Definition 5** (Equivalence). We speak of data equivalence when we cannot distinguish them from one another after normalization:

$$\forall x, y \in \omega : x \equiv y$$

$$\iff \begin{cases} x^\tau = y^\tau \\ \|\nu(x)\| = \|\nu(y)\| \\ n \leftarrow \|\nu(x)\|, \forall i := [1..n] : \\ \nu(x)[i] = \nu(y)[i] \end{cases} \quad (2)$$

In other words, two input data are said equivalent if they share the same data type and their normalized versions are identical.

Normalization could be applied differently depending on the data type  $d^\tau$ .

The following are the transformations that might be applied to an input data source  $d^\zeta$ :

- Replacement of punctuation and special characters by spaces, eg.

$$\left. \begin{array}{l} - \\ \cdot \\ \cdot \\ * \\ \text{etc.} \end{array} \right\} \mapsto \backslash s$$

- Trim and extraction of excess spaces, eg.

$$\backslash sMy\backslash s\backslash sdata \mapsto My\backslash sdata = My \text{ data}$$

- Removal of accents, eg.  $\acute{e} \mapsto e$ ,  $\tilde{n} \mapsto n$ ;
- Capitalization, eg.  $Cyril \mapsto CYRIL$ ;
- Dictionary substitution eg.

$$\left. \begin{array}{l} AVE \\ AVENUE \\ AVN \\ AVNU \end{array} \right\} \mapsto AV \quad \left. \begin{array}{l} F \\ Madam \\ Ms \\ Mrs \end{array} \right\} \mapsto 2 \text{ etc.}$$

In some case, normalization could transform the source data to a single space which in turn would lead to  $d_i$  being discarded.

**Example 1.** Let

$$\left\{ \begin{array}{l} d_1^\zeta = 1600, \text{ Pennsylvania Avenue NW.} \\ d_2^\zeta = 1600 \text{ PENNSYLVANIA AV NW} \end{array} \right.$$

be two source data of the same data type:

$$d_1^\tau = d_2^\tau = \text{address4,}$$

then we have:

$$\nu(d_1^\zeta) = \nu(d_2^\zeta) = d_2^\zeta \implies d_1 \equiv d_2$$

As  $d_1$  and  $d_2$  are equivalent, only one of them would be kept in the set of normalized data  $\mathcal{D}$  to pass onto the recombination step.

## 2.2 Recombination

Once the data will be irreversibly ciphered into the final *Empreinte Sociométrique*, we need to maximize the chance to get a match when using it, the goal of the recombination function  $\rho$

is to provide as many representations as possible of the data.

Each data type may have dozens or even hundreds of possible combinations so we won't detail them here. Instead, we will use an example.

**Example 2.** Let the vector  $\mathbf{d}$  be the input data:

$$\{(\text{gender}, \text{Mr.}), (\text{firstname}, \text{John}), (\text{middle}, \text{F.}), (\text{lastname}, \text{Kennedy}), (\text{streetNumber}, 1600), (\text{streetName}, \text{Pennsylvania Avenue}), (\text{city}, \text{Washington}), (\text{zip}, \text{DC 20500})\}$$

Let  $\mathcal{D} \leftarrow \nu(\mathbf{d})$  be a set of normalized data:

$$\{1, \text{JOHN}, \text{F}, \text{KENNEDY}, 1600, \text{PENNSYLVANIA AV}, \text{WASHINGTON}, \text{DC 20500}\}$$

Then  $\mathcal{R} \leftarrow \rho(\mathcal{D})$  might yield the following additional values:

- 1 JOHN;
- JOHN F;
- 1 JOHN F;
- JOHN KENNEDY;
- 1 JOHN KENNEDY;
- JOHN F KENNEDY;
- 1 JOHN F KENNEDY;
- 1600 PENNSYLVANIA AV;
- WASHINGTON DC 20500;
- 1600 PENNSYLVANIA AV WASHINGTON DC 20500;
- JOHN KENNEDY 1600 PENNSYLVANIA AV WASHINGTON DC 20500;
- 1 JOHN KENNEDY 1600 PENNSYLVANIA AV WASHINGTON DC 20500;
- JOHN F KENNEDY 1600 PENNSYLVANIA AV WASHINGTON DC 20500;
- 1 JOHN F KENNEDY 1600 PENNSYLVANIA AV WASHINGTON DC 20500.

$$\implies \|\mathcal{R}\| = 22$$

Once ciphered, these 22 recombined contacts will form the basis of the corpus.

## 2.3 Ciphering

Lorem ipsum ...

## 2.4 Corpus

Lorem ipsum ...

## 2.5 Signature

Lorem ipsum ...

## 2.6 Output

Lorem ipsum ...

# CONTENTS

<b>I</b>	<b>Introduction</b>	<b>1</b>
<b>II</b>	<b>Formal Description</b>	<b>1</b>
1	General Algorithm . . . . .	1
2	Step-by-Step Process . . . . .	2
2.1	Normalization . . . . .	2
2.2	Recombination . . . . .	3
2.3	Ciphering . . . . .	3
2.4	Corpus . . . . .	3
2.5	Signature . . . . .	4
2.6	Output . . . . .	4

# REFERENCES

- [1] Cyril Dever. *Système de traitement d'une base de données personnelle par un opérateur extérieur*, patent pending FR1905778, 2019.

## APPENDIX

**Table 1:** *Input data types*

Code	Description	Examples
gender	Title or gender	<i>M, Mr., 1, ...</i>
firstname	First name or given name	<i>John</i>
middle	Middle initials or other names	<i>J., John</i>
birthname	Birth name	<i>Kennedy</i>
lastname	Last name or married name	<i>Kennedy</i>
suffix	Suffix	<i>Jr.</i>
birthdate	Date of birth	
birthplace	Place of birth	
addresses	List of postal address	(see Table 2)
aliases	List of aliases	(see Table 3)
emails	List of e-mail addresses	(see Table 4)
ids	List of official IDs	(see Table 5)
mobiles	List of mobile phones	(see Table 6)
phones	List of telephone numbers	(see Table 7)
socials	List of social media	(see Table 8)
updated	Unix timestamp of collect	<i>1544529071</i>

**Table 2:** *Address data type*

Field	Definition	Possible values (or examples)
type	Address type	"birth" ∨ "home" ∨ "work"
address2	Additional name	<i>c/o Mme Dupont</i>
address3	Additional address	<i>Apt. 123</i>
address4	Street number and name	<i>1600 Pennsylvania Ave NW</i>
address5	PO Box or locality	<i>BP 987</i>
address6	City and ZIP code	<i>Washington, DC 20500</i>
address7	International destination	<i>U.S.A.</i>
city	City	<i>Washington</i>
country	Country	<i>United States of America</i>
fullAddress	Full address	<i>1600 Pennsylvania Ave NW, Washington, DC 20500</i>
streetName	Street name	<i>Pennsylvania Ave NW</i>
streetNumber	Street number	<i>1600</i>
zip	ZIP code	<i>DC 20500</i>
updated	Time of collect	<i>1544529071</i>

**Table 3:** *Alias data type*

Field	Definition	Possible values (or examples)
type	Alias type	"commonname" ∨ "identity" ∨ "np" ∨ "pn" ∨ "pseudo" ∨ "tnp" ∨ "tpn"
value	Alias value	<i>John John</i>
updated	Time of collect	1544529071

**Table 4:** *E-mail data type*

Field	Definition	Possible values (or examples)
type	E-mail type	"business" ∨ "personal" ...
value	E-mail address	<i>john@john.com</i>
isHash	If is already hashed	true ∨ false
engine	Hashing algorithm	"blake2" ∨ "md5" ...
updated	Time of collect	1544529071

**Table 5:** *ID data type*

Field	Definition	Possible values (or examples)
type	ID type	"id" ∨ "cb" ∨ "passport" ∨ "registration" ∨ "serial" ∨ "ss" ∨ "udid"
value	ID value	<i>1234567890abcdef</i>
isHash	If is already hashed	true ∨ false
engine	Hashing algorithm	"blake2" ∨ "md5" ...
updated	Time of collect	1544529071

**Table 6:** *Mobile phone data type*

Field	Definition	Possible values (or examples)
type	Mobile phone type	"business" ∨ "personal" ...
value	Mobile phone number	+33 (0) 623 456 789
isHash	If is already hashed	true ∨ false
engine	Hashing algorithm	"blake2" ∨ "md5" ...
format	Format	" +dd (d) ddd ddd ddd"
updated	Time of collect	1544529071

**Table 7:** *Telephone data type*

Field	Definition	Possible values (or examples)
type	Telephone type	"business" ∨ "personal" ...
value	Phone number	+33 (0) 123 456 789
isHash	If is already hashed	true ∨ false
engine	Hashing algorithm	"blake2" ∨ "md5" ...
format	Format	" +dd (d) ddd ddd ddd"
updated	Time of collect	1544529071

**Table 8:** *Social media data type*

Field	Definition	Possible values (or examples)
type	Alias type	"facebook" ∨ "linkedin" ∨ "twitter" ∨ "youtube" ...
value	Alias value	<i>@john</i>
updated	Time of collect	1544529071