Compact Identity Footprint

Cyril Dever
Edgewhere

September 23, 2020

Abstract

We define a compact identity footprint for anyone who needs to have compact pseudonymized representation of an identity. It should be used in conjunction with any other type of data footprint to associate the latter with its rightful owner.

I. Introduction

Sometimes, one needs a way to compare data without having access to its source. Usually, the use of any cryptographic hash function would suffice. But what if you also need to be sure of its ownership?

The present document describes a simple yet effective way to build such a simple "owned" footprint. We call it the Peeel™.

II. Quick Description

The Peeel™ algorithm is used to create a unique string from contact data that may eventually serve as some proof of ownership to another data with which it would be associated.

It follows the simple rules below:

- Each source data must first be normalized according to the *Empreinte Sociométrique* standards [1];
- The normalized data is then hashed using the `SHA-256` algorithm applied twice;
- The hashed data is then concatenated in the exact order of the complementaries code, eg. hashed date of birth then hashed firstname for a `"dob+firstname"` code;
- This concatenation is next itself hashed using the `SHA-256` algorithm twice to form the final hash data.

The final result is a tuple of the hexadecimal string representation of the returned hash and the set of codes used, eg.

$$(\texttt{29023e[...]e9a2}, \{\texttt{firstname}, \texttt{dob}\})$$

Definition 1 (Complementary Data)**.** We call complementary data χ_i of an identity i each different contact data that would be used to build his Peeel. It is defined by the tuple of its code χ^c and its associated data source χ^d, eg.

$$\chi_i \leftarrow \begin{cases} \chi_i^c := \texttt{firstname} \\ \chi_i^d := \text{Cyril} \end{cases}$$

As of this version, the available complementary data codes could be one of the following:

- `dob`: a date of birth (respecting the ISO format, ie. `YYYYMMDD`);
- `gender`: 1 for male, 2 for female, or 0 for unknown or indeterminate;
- `firstname`: a first name;
- `lastname`: a last name;
- `middle`: the middle names or initials (eg. the *F.* in *John F. Kennedy*).

We define this set of available codes as **C**.

III. Algorithm

Let $N_{ES} : \omega \to \omega$ be the normalizing function through the Empreinte Sociométrique standard for the input data[1], and $\mathfrak{h}()$ the cryptographic hashing function[2].

And let $\chi_1, \chi_2, \ldots, \chi_n$ be an ordered set of complementary data.

Algorithm 1 describes how a Peeel is built.

[1] for example, our Javascript `es-normalizer` library: `https://npmjs.com/package/es-normalizer`

[2] we use `SHA-256` for its wide adoption in all the major browsers

Algorithm 1: Peeel™ algorithm

Input: A set of complementary data $\{\chi_1, \chi_2, \ldots, \chi_n\}$
Output: The hashed data and its set of used codes, or an error

initialize the set of normalized hashed items $\mathcal{H} \leftarrow \emptyset$, and the ordered set of kept codes $\mathcal{C}_k \leftarrow \emptyset$;
for $i \leftarrow 0$ **to** n **by** 1 **do**
  if $\chi_i^c \notin \boldsymbol{C}$ **then**
    continue;
  $\mathcal{H} \leftarrow \mathfrak{h}\left(N_{ES}(\chi_i^d)\right)$;
  $\mathcal{C}_k \leftarrow \chi_i^c$;
if $|\mathcal{H}| = 0 \vee |\mathcal{K}| \neq |\mathcal{H}|$ **then**
  throw *invalid input*
initialize the concatenation string S;
for $i \leftarrow 0$ **to** $|\mathcal{H}|$ **by** 1 **do**
  $S \leftarrow S \parallel \mathcal{H}_i$;
build the hexadecimal string $hexStr := \left(\mathfrak{h}(S)\right)_{16}$
return $(hexStr, \mathcal{C}_k)$;

The returned set of used codes should be in the same order than the input set of complementary data codes. In other words, if the input set is totally valid, we should have:

$$\mathcal{C}_k := \{\chi_1^c, \chi_2^c, \ldots, \chi_n^c\}$$

IV. Usage

The main usage of the Peeel™ algorithm is in association with another data to make it a kind of proof of ownership of this other data in full respect of data privacy regulations such as GDPR or CCPA.

For example, in our Consent Management Blockchain[2], we use it to link the consented data with the recorded transaction. That way, whenever a data is consented, we can make sure that the person who consents is the current rightful owner of the data and that further use of the data by an eventual previous owner can't be possible anymore. And if we want proof of ownership, we just need to implement some sort of questionnaire to rebuild the Peeel (without even knowing anything about it except for the codes and therefore the kind of question to ask)[3].

References

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

[2] Cyril Dever. *Deferred Resolution Consensus Protocol on Double-Helix Double-Circuit Blockchains*, 2020.

[3] You might want to check out our Fruuut™ library: `https://github.com/cyrildever/fruuut`, to see our last implementation of such a questionnaire.