Journal of Information Security and Applications 74 (2023) 103478

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Enabling anonymized open-data linkage by authorized parties

Francesco Buccafurri *, Vincenzo De Angelis, Sara Lazzaro

University of Reggio Calabria, Via dell'Università, 25, Reggio Calabria, 89124, Italy

ARTICLE INFO

Keywords:
Open data
eIDAS
Anonymity
Record linkage

ABSTRACT

Nowadays, many entities collect useful information about users, in order to implement the provided service, and publish them as open data. To prevent privacy leakage, data are often anonymized prior to publication. Unfortunately, anonymization strongly hinders data linkage, which can be very useful for analysis purposes instead. In this paper, we deal with the above problem, by proposing a technique that enriches anonymized open data with pseudo-random labels. This way, some authorized parties (i.e., the analysts) are enabled to link data regarding the same user coming from different sources. Instead, for non-authorized people, labels do not carry any information, thus not introducing additional privacy threats with respect to original open data. In other words, our solution allows us to recover linkage capabilities on anonymized open data, thus enabling more powerful data exploitation. Indeed, the linked open data paradigm, involving both the public sector and business, is recognized as one of the most promising approaches for boosting societal growth. To offer a concrete solution, we refer to an existing open-data standard and we implement the protocol through a SAML-based SSO framework adhering to the eIDAS regulation.

1. Introduction

In the current digital era, data represent very valuable assets, because they are the basis for strategic tasks and decisions, in various fields, such as business, e-government, e-health, research, and so on. For this reason, the open-data paradigm is assuming a very relevant role in our society [1]. Open data consist of information that can be accessed, used, and shared by anyone [2].

In the scientific literature, numerous papers witness the benefits derived from the use of open data [2,3]. Indeed, open data can improve the efficiency of public services [4,5], but also produce economic growth in the private sector [6].

Despite all the benefits coming from the exploitation of these data in different scenarios, many privacy issues may arise. To prevent privacy leakage [7–9], data are often anonymized prior to publication.

In the literature, several proposals are available with the aim to anonymize the data published by a source and prevent the linkage with the real identity of users [10–13].

When dealing with open data published from different sources, it becomes relevant capturing possible links between data (belonging to the same user) to perform more powerful and efficient analysis.

Unfortunately, anonymization strongly hinders data linkage. Even though, in principle, linking attempts can be made on anonymized databases (for example, by performing composition attacks [14,15]), they do not guarantee the effectiveness of the results in terms of completeness.

Although this may be considered a desirable feature from a privacy perspective, it considerably limits the effectiveness of data analysis.

The aim of this paper is to propose a mechanism to recover the full linking capability when anonymized techniques are applied prior to publication. On the other hand, it appears unnecessary and potentially dangerous to disclose such a linkage to other than authorized parties (i.e., the analysts).

The idea of our solution is to associate the data with some pseudo-random labels that do not carry any information for non-authorized parties. Conversely, through the knowledge of a secret, the analysts can link the data by exploiting such labels.

Therefore, our solution does not introduce any additional privacy threats with respect to the original anonymized open data, concerning their public access.

It is worth noting that our solution is orthogonal with respect to the techniques used to anonymize data, which is a problem out of the scope of this paper.

Another contribution of the paper is the implementation of the proposed solution leveraging widely-adopted standards and adhering to the European regulation eIDAS [16]. The proposed solution is integrated into the Single-Sign-On authentication framework [17] that allows users to authenticate with different service providers by using a single set of credentials. In particular, we refer to the (eIDAS-compliant) SAML-based SSO authentication and show how it can be extended to support our proposal.

* Corresponding author.
E-mail address: bucca@unirc.it (F. Buccafurri).

https://doi.org/10.1016/j.jisa.2023.103478

Available online 31 March 2023

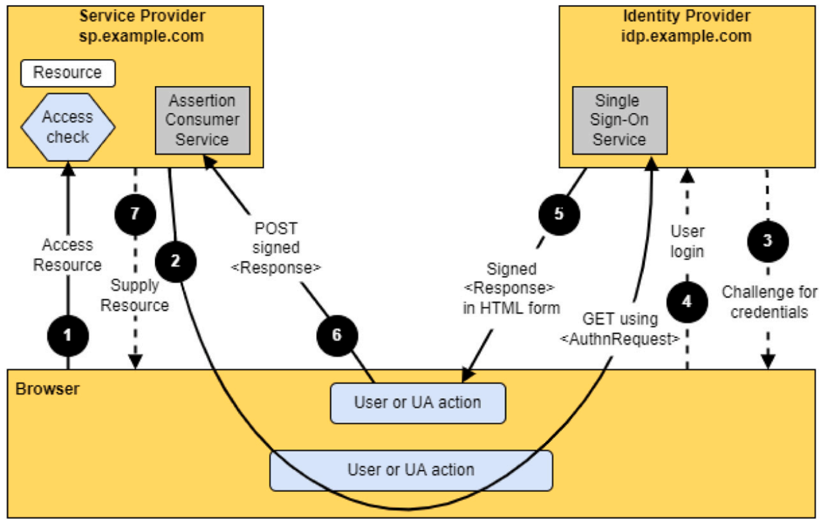

Fig. 1. SSO SAML-based authentication procedure.

SAML 2.0 [18] is an XML-based standard for the exchange of secure authentication and authorization messages. It is widely used in government and enterprise environments when the Single Sign-On (SSO) approach is adopted.

The implementation of the solution, along with a case study, is also provided to witness the feasibility of our proposal.

The structure of the paper is the following. In Section 2, we recall some background notions about open data and the SAML-based SSO framework. Then, in Sections 3 and 4, we describe our proposal. Its implementation and a case study are discussed in Section 5. In Section 6, we analyze the security aspects of our proposal. The related literature is discussed in Section 7. Finally, in Section 8, we draw our conclusions.

2. Background

Through this section, we provide some background notions about open data and the SAML-based SSO framework. In particular, Section 2.2 describes in detail the authentication procedure that is also leveraged by our protocol described in 3.

2.1. Open data

Open data consist of information that can be accessed, used, and shared by anyone. The only constraints in sharing them are represented by the obligation to acknowledge the source and to use the same type of license under which they had been previously released. In many contexts, users interact with several service providers. From these interactions, the service providers can draw valuable data about users. These data, properly pre-processed, can be published as open data so that they can be analyzed by other parties.

As a best practice, Tim Berners-Lee introduces five levels [19] for the definition of the format in which open data should be published. Each additional level presumes the data meet the criteria of the previous levels. The first level refers to data made available on the web in any format (not necessarily machine-readable) under an open license. The second level refers to machine-readable structured data (such as Excel). The third level requires that the data are not in a proprietary format (for instance CSV instead of Excel). Level 4 requires the adoption of open standards from the W3C (such as RDF and SPARQL). Finally, Level 5 refers to *Linked Open Data* [20,21]. This level requires that machine-readable data coming from different sources can be linked to perform much more interesting analyses, compared to data coming from a single source.

In the rest of the paper, we will refer to open data published in a level 5 format.

2.2. eIDAS and SAML 2.0

The eIDAS regulation aims to "provide a common normative basis for secure electronic interactions between citizens, businesses and public administrations and at increasing the security and effectiveness of electronic services and e-business and e-commerce transactions in the European Union" [16]. In this paper, we focus on the eIDAS authentication framework for the management and verification of citizens' digital identities. This framework is based on the concept of interoperability in such a way that the member states recognize the digital identities issued by other member states to promote cross-border cooperation.

Two standards are mainly adopted to implement the eIDAS authentication framework: SAML 2.0 [18] and OpenID Connect [22]. In this paper, we refer to the former, which is a standard largely used in government and enterprise environments especially when the single Sign-On (SSO) approach is adopted. SSO is an authentication method that allows users to authenticate with multiple services by using a single set of credentials.

SAML 2.0 is an XML-based standard for the exchange of secure authentication and authorization messages. There are three main actors:

- Users: they are associated with a digital identity registered with an identity provider. They need to prove such an identity to a service provider to obtain a service.
- Service provider: it provides a service to users after obtaining guarantees about their digital identity.
- Identity provider: it manages users' digital identities and provides the service provider with an assertion certifying each digital identity.

We now describe the SAML authentication procedure that involves the above-mentioned actors. This procedure performs in several steps reported in Fig. 1, in which the browser represents a user.

1. The user asks the service provider for a resource (service).
2. Since the user is not authenticated, the service provider generates an `Authentication request` that is forwarded to the identity provider by the user.
3. The identity provider asks the user for their credentials.
4. The user authenticates with the identity provider.
5. If the authentication is successful, the identity provider generates a `Response` containing an `Assertion` that certifies the success of the authentication. This assertion is digitally signed and forwarded (through the user) to the service provider.

6. The service provider checks the digital signature and the validity of the assertion.
7. If the previous check is successful, the service provider supplies the required resource.

Observe that the user can leverage the same credentials to authenticate with a different service provider. Indeed they are provided to the identity provider and not directly to the service provider. This is exactly the goal of SSO.

3. Problem formulation and notation

In this section, we introduce the notation we use in the rest of the paper. We denote by $\{S_1, \dots, S_z\}$ a set of service providers. Each of these providers offers a certain service to users. For each interaction of a user with a service provider S_i, S_i generates a set of data associated with the user in this interaction. We denote by $D_i^j(t)$, the set of data generated by S_i in the tth interaction with the user j.

We denote by α a function that takes as input the real identity of j and the data $D_i^j(t)$ (generated by S_i in the tth interaction with j) and returns as output a label $P_i^j(t)$.

This label is associated with $D_i^j(t)$ and the pair $E_i^j(t) = \langle P_i^j(t), D_i^j(t)\rangle$ represents an entry of the database D_i stored by S_i.

D_i will be published by S_i as open data, thus making it publicly available so that it can be freely used for different purposes. The function α aims to hide the real identity of a user. Indeed, the label $P_i^j(t)$ should not be linkable to the real identity of j even knowing all the entries of D_i. Moreover, for different entries associated with the same user j in different interactions, these labels should not be linkable between them. A trivial way to implement α is to generate a random number for each $D_i^j(t)$. However, this approach does not meet our requirements since it prevents any linkage of data from any party even though authorized. Then, we want the result of the function α to appear random for any entity except for some authorized parties.

Observe that, another problem (orthogonal to our proposal) is about the fact that the labels obtained through the function α do not prevent the re-identification of the users if the entries of the database contain other information (i.e., *quasi-identifiers*) that can be associated with the real users' identities through background knowledge [23]. Indeed, the remaining information of each entry $E_i^j(t)$ (i.e., $D_i^j(t)$) can be used to re-identify individuals by linking or matching the data with other data or by examining unique features found in the released data [23].

Then, before being published, these data must undergo an anonymization process to make them compliant with privacy regulations.

As discussed in Section 7, advanced privacy-preserving techniques must be applied to the data. In the following, we denote by δ the overall transformations applied to D_i to make the data harder to de-anonymize. We denote by $\overline{D_i} = \delta(D_i)$, the result of the anonymization function that will be eventually published by S_i.

Observe that, since the results of α are not identifiers or quasi-identifiers (they appear as random values not linkable among them), it is safe to assume that the function δ preserves such values without any modification. Then, after the anonymization process, the entries of $\overline{D_i}$ will be in the form $\overline{E_i^j}(t) = \langle P_i^j(t), \overline{D_i^j}(t)\rangle$.

The objective of this paper is to design a solution that implements the function α. We summarize the requirements of this function.

- The entries published by a service provider associated with the same user can be linked together only by some authorized parties (and the provider itself), through the label obtained by the function α.
- The entries published by a service provider (associated with a user) can be linked with the entries published by another provider (associated with the same user) only by authorized parties, through the label obtained by the function α.

Observe that the second requirement includes that also a service provider cannot link the entries (associated with the same user) that it publishes with the entries published by any other provider.

We formally define the above properties in Section 6.

As a final remark, we observe that if the above properties are satisfied, then the function α does not introduce any additional privacy leakage with respect to non-authorized entities. On the other hand, it allows the authorized entities to perform the linkage of the data.

4. The proposed protocol

In this section, we propose a solution for implementing the function α that enables the open-data linkage.

We distinguish two phases in our protocol.

4.1. Interaction between a user and a service provider

We consider four actors:

- A user j,
- A service provider S_i,
- An identity provider IP,
- A set of analysts $\mathcal{A}_i$ interested in the data published by S_i.

The first three mentioned actors are the three parties that interact in a classical SSO approach as described in Section 2.2.

The service provider has the faculty of collecting data from the interactions with the users. Such data, properly anonymized, will be published in an open-data format so that they will be publicly available to any other external party (i.e., parties not directly involved in the authentication process). To be concrete, we refer to an identity provider adhering to the eIDAS regulation as described in Section 2.2. However, our solution can be easily adapted to any different SSO-based approach.

We also define a fourth actor, i.e., a set of analysts $\mathcal{A}_i$ that are authorized to link the data published by S_i. Moreover, if some of these analysts are also authorized by another service provider S_k, they will be able to link the data published by S_i with the data published by S_k. This will be discussed in the next section. We assume that, all the analysts in $\mathcal{A}_i$ share a secret X_i associated with the service provider S_i.

We consider the tth interaction between j and S_i.

The result of this interaction will be a pair $E_i^j(t) = \langle P_i^j(t), D_i^j(t)\rangle$. We recall that $D_i^j(t)$ are the data associated with j by S_i during this interaction. In this section, we show how to compute the label $P_i^j(t)$ to associate with $D_i^j(t)$.

In our approach, j authenticates with the service provider S_i, after interacting with IP via a SAML-based authentication.

However, as reported in Fig. 2, our solution requires a modification of the SAML authentication procedure. Indeed, in our proposal, we need to include, in the `Assertion` message (step 5 of Fig. 1) the following information:

- an order number N^j. This value represents the number of authentications performed (through the identity provider IP) so far by the user j with all the service providers. In other words, N^j will be incremented by one every time a user is successfully authenticated through IP, regardless of the service provider to which j is willing to connect. For example, if j authenticates three times with S_i and four times with S_k, N^j is equal to 7 regardless of the order of the authentications.
- a value $Y^j = MAC(I^j, Secr^j)$, where MAC represents a secure message authentication code applied to an identifier I^j (associated by IP with the real identity of j) with key $Secr^j$, that is a secret owned by IP associated with j. This secret will prevent an external party from discovering I^j through a dictionary attack performed on Y^j. Moreover, as Y^j is the output of a hash function, no collision can be found. Therefore, Y^j is uniquely associated with the user j. Finally, observe that two different

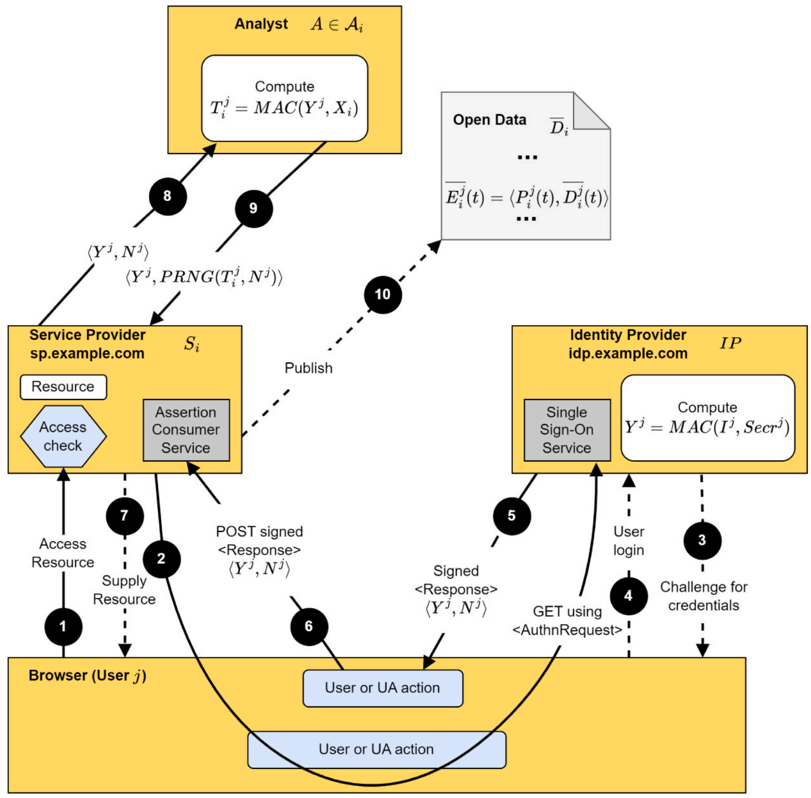

Fig. 2. SSO SAML-based proposed solution.

service providers will receive the same Y^j when the same user j interacts with them. This is on the basis of the procedure performed by the analysts allowing the data linkage.

Once S_i receives the assertion containing $\langle Y^j, N^j \rangle$, the following steps are performed:

- S_i sends the pair $\langle Y^j, N^j \rangle$ to each analyst A in $\mathcal{A}_i$.
- Each analyst A in $\mathcal{A}_i$ maintains a hash table H_i (for the service provider S_i) that associates each value Y^j, received by S_i, with a list L_i^j of numbers. Specifically, when A receives the pair $\langle Y^j, N^j \rangle$, it adds N^j to the list L_i^j associated with Y^j in H_i.
- S_i chooses randomly an analyst A^* belonging to $\mathcal{A}_i$, to obtain a label $P_i^j(t)$ to associate with $D_i^j(t)$.

A^* proceeds as follows:

- A^* computes $T_i^j = MAC(Y^j, X_i)$,
- A^* uses T_i^j as seed of a PRNG (pseudorandom number generator), and computes the value $PRNG(T_i^j, N^j)$, denoting the N^jth number obtained by the PRNG;
- A^* sends S_i the pair $\langle Y^j, PRNG(T_i^j, N^j) \rangle$.

$PRNG(T_i^j, N^j)$ is just the label $P_i^j(t)$ to associate with $D_i^j(t)$.

Then, S_i locally stores the entry $E_i^j(t) = \langle P_i^j(t), D_i^j(t) \rangle = \langle PRNG(T_i^j, N^j), D_i^j(t) \rangle$ in the database D_i.

Periodically, S_i applies the function δ to D_i and publishes the resulting anonymized database $\overline{D_i}$ as open data.

4.2. Open-data linkage

In this section, we show how the analysts link the data published in the anonymized databases. In the following, we distinguish two cases. **Linkage in the same database.** In the first case, we consider an analyst $A \in \mathcal{A}_i$ that wants to link the entries belonging to the same users in an anonymized database $\overline{D_i}$, published by the service provider S_i.

We recall that $\overline{D_i}$ contains a set of entries in the form $\overline{E_i^j}(t) = \langle P_i^j(t), \overline{D_i^j}(t) \rangle$ for some j and t.

A performs as follows:

- for each key Y^* of the hash table H_i, A retrieves the associated list L_i^* and computes $T_i^* = MAC(Y^*, X_i)$.
- for each N^* in L_i^*, A computes $\widehat{P} = PRNG(T_i^*, N^*)$.
- A finds the entry of $\overline{D_i}$ such that $P_i^j(t) = \widehat{P}$ and replaces this value with Y^*.

The above procedure is summarized in Algorithm 1.

Algorithm 1: Linkage of open data in $\overline{D_i}$.

for $Y^* \in H_i$ **do**
  $L_i^* \longleftarrow H_i[Y^*]$;
  $T_i^* \longleftarrow MAC(Y^*, X_i)$;
  for $N^* \in L_i^*$ **do**
    $\widehat{P} \longleftarrow PRNG(T_i^*, N^*)$;
    for $E = \langle A, B \rangle \in \overline{D_i}$ **do**
      if $A == \widehat{P}$ **then**
        $A \longleftarrow Y^*$

At the end of this procedure, A obtains a modified database $\widehat{D_i}$ such that all the entries with the same first component belong to the same user. Thus, the linkage is performed.

Linkage between two different databases. In the second case, we consider an analyst $A \in \mathcal{A}_i \cap \mathcal{A}_k$ that wants to link the entries belonging to the same users in the anonymized database $\overline{D_i}$ (published by S_i) and in the anonymized database $\overline{D_k}$ (published by S_k). In other words, A wants to join the two databases and link all the entries belonging to the same users.

A performs as follows:

- A invokes Algorithm 1 on the database $\overline{D^i}$ and obtains $\widehat{D_i}$.
- A invokes Algorithm 1 on the database $\overline{D^k}$ and obtains $\widehat{D_k}$.
- A joins all the entries in $\widehat{D_i}$ and $\widehat{D_k}$ where the first component is the same, i.e., the entries belonging to the same user.

5. Case study and implementation

Through this section, we provide the implementation of the protocol described in Section 4 and show how it works in a case study.

Our implementation consists of four modules that correspond to the actors of the protocol, i.e., the user, the identity provider, the service provider, and the analyst. The user module is simply represented by a web browser. The identity provider module is based on Keycloak [24], an open-source JAVA implementation of an identity management system that enables SSO authentication. To implement the functions described in Section 4, we properly modified the `saml-core.jar` library, by adding the components we need and by intervening, in particular, on the SAML `assertion`. We will provide further details in the sequel of the section. Finally, the service provider and the analyst modules have been implemented from scratch through Servlet and JSP technology [25]. As the format for the open data, we choose JSON-LD [26], a lightweight Linked Data format recommended by W3C. It implements the level 5 format for open data described in Section 2.1.

The case study considered is the following.

We have an identity provider IP, two service providers S_i and S_k, and an analyst $A \in \mathcal{A}_i \cap \mathcal{A}_k$ interested in linking the data from both S_i and S_k. Suppose S_i is an online pharmacy that maintains a database D_i in which each entry is associated with a user's order containing some sensitive information such as the list of the purchased medicines.

Concerning S_k, it is an online grocery shop that maintains a database D_k in which each entry keeps track of the products purchased by a user.

Observe that, in both D_i and D_k the same user may appear more times.

The goal of A is to link D_i and D_k (after they are published in anonymous form) to infer some information, i.e., whether there is a correlation between the medicines purchased by a user (and then the diseases they suffer from) and the products they purchased from the online store.

5.1. Interaction between a user and the service providers

Consider a user named John Smith (j) interacting with both S_i and S_k.

As the first interaction, j authenticates with S_i through IP to buy the medicines `MedA` and `MedB`. IP computes the values Y^j and N^j, as described in Section 4.1, and sends them to S_i. In our implementation, IP stores the (John's) secret $Secr^j$ and the value N^j that counts the number of authentications performed so far by John (with all the service providers). Suppose the secret of John is $Secr^j$ = `superSecretPassword` and $N^j = 0$ (i.e., this is the first authentication of John). Since the computation of Y^j requires an identifier of John maintained by IP, we used, for simplicity, the Keycloak username of John, say `johnSmith20`. Finally, we implemented the MAC function through $HMAC$ [27] based on the cryptographic hash function SHA256.

In the listing of Fig. 3, we show a fragment of code to compute $\langle Y^j, N^j \rangle$ and set them in the SAML `Assertion` for the service provider. This code has to be included in the class `org.keycloak.saml.processing.api.saml.v2.response.SAML2Response` of the `saml-core.jar` library. Observe that the instruction in Line 20 sets the pair $\langle Y^j, N^j \rangle$ in field `SubjectID` of SAML `Assertion` in place of the standard username.

With the values set as above, S_i will receive the pair ⟨`d94fe9ff76414b9e742819635f7dccf5fddd03c45e201ab34976f2cd9b4459a7,1`⟩.

Such a pair is retrieved by a service provider (implemented through a Servlet) with the instruction `String pair=request.getUserPrincipal().getName()+"-"+UUID.randomUUID().toString().replace("-", "")`.

At this point, S_i forwards such a pair to all the analysts in A_i (among which A). Moreover, it selects $\overline{A} \in \mathcal{A}_i$ to obtain the pseudonymous to associate with John's data.

$\overline{A}$ computes $T_i^j = MAC(Y^j, X_i)$, where X_i is a secret shared among all the analysts in $\mathcal{A}_i$ and associated with S_i. Suppose X_i = `Analyst-Secret`. Again, we chose $HMAC$ to implement the MAC function, then resulting in T_i^j = `13715fb857d317962073856cbedbbf417c9d68eb1fe411d6713f260b7ec8af4a`. To obtain the pseudonymous to be associated with the data, A_i needs to compute $PRNG(T_i^j, N^j) = PRNG(T_i^j, 1)$. We implemented the PRNG through a cryptographically strong random number generator (CRNG) [28]. In particular, we relied on the Java class `SecureRandom` and chose the algorithm `SHA1PRNG`. The complete code implemented in the analyst module is reported in the listing of Fig. 4.

The result of this computation is $PRNG(T_i^j, 1)$ = `1807256804637968330` that is provided, along with Y^j, to S_i.

At a given point, S_i wants to publish the database D_i with the entries so far collected. In Table 1, we represent the database D_i used in this case study.

Observe that, the second row of Table 1 corresponds to the interaction of John described above.

Before publishing D_i, the function δ has to be applied so that the data are anonymized. In this case study, we applied the k-anonymity technique [10].

Specifically, the attribute `Name` is an identifier while `Date of Birth`, `Gender`, `Domicile` are quasi-identifiers. `Label` is a non-identifying and non-sensitive attribute while `Products` is a non-identifying sensitive attribute.

By applying the k-anonymity technique (with $k = 2$), we obtain the anonymized database $\overline{D_i}$ reported in Table 2.

Observe that all the values of the attribute `Name` are suppressed. The exact values of the attribute `Date of Birth` are replaced by intervals and the exact values of the attribute `Domicile` are replaced with a broader region. The other values of the other attributes are unaltered. Through this procedure, there are at least two entries of $\overline{D_i}$ with the same values of the quasi-identifier attributes, i.e., `Gender`, `Date of Birth`, `Domicile`.

At this point, S_i can publish $\overline{D_i}$ as open data. In this case study, we consider the JSON-LD format for the open data resulting in a JSON file for each entry. For example, the (anonymized) entry associated with the order of John (second row of Table 2) is shown in the listing of Fig. 5.

Therein, we refer to the Schema.org vocabulary [29], managed by a collaborative community with the aim to create, maintain, and promote schemas for structured data on the Internet. This way, our solution maintains full interoperability between data generated by different service providers. In this example, we have an object that represents an order containing information about the person requesting it (i.e., type Person) and the list of products included in the order (i.e., type Products).

A procedure, similar to the one described above, is followed by S_k when publishing the database D_k, represented in Table 3.

In this example, we suppose the second and seventh rows of Table 3 correspond to two orders performed by John with S_k (the two rows have the same credit card number, date of birth, and domicile). In the first order, he purchased three products `ProdA`, `PodB`, `ProdC`. We suppose this represents the second interaction (i.e., $N^j = 2$) made by John. In the second order (third interaction made by John, i.e $N^j = 3$),

```java
byte[] Y= null; Integer N;
try {
 File file = new File("PATH_TO_SECRET_KEY");
 BufferedReader br = new BufferedReader(new FileReader(file));
 String key=br.readLine();
 Mac mac = Mac.getInstance("HmacSHA256");
 SecretKeySpec secretKeySpec = new SecretKeySpec(key.getBytes(), "HmacSHA256");
 mac.init(secretKeySpec);
 Y = mac.doFinal(idp.getNameIDFormatValue().getBytes());

 file = new File("PATH_TO_N");
 br = new BufferedReader(new FileReader(file));
 String n=br.readLine();
 N= Integer.parseInt(n)+1;
 FileWriter myWriter = new FileWriter("PATH_TO_N");
 myWriter.write(String.valueOf(N));
 myWriter.close();

 String pair=encodeHexString(Y)+"-"+String.valueOf(N);
 nameIDType.setValue(pair);
 }catch (Exception e) {
 }
```

Fig. 3. Fragment of code to be integrated into the library `saml-core.jar` included in Keycloak.

Table 1
Database D_i collected by S_i.

Label	Name	Gender	Date of Birth	Domicile	Products
516702555509767784	Jimmy Collins	Male	1933-07-08	Austin (Texas)	[MedC]
1807256804637968330	John Smith	Male	1964-11-04	Los Angeles (California)	[MedA, MedB]
460853062988418469	Jennifer Johnson	Female	1993-09-12	Henderson (Nevada)	[MedC, MedD]
79983861162328468	Alex Garcia	Male	1966-06-14	San Diego (California)	[MedE]
2176216674885739653	Kate Williams	Female	2004-01-30	Las Vegas (Nevada)	[MedC]
5541821146178023331	Ricky Stewart	Male	1934-04-06	Houston (Texas)	[MedL]
3745388544143800788	Kelly Morgan	Female	1998-12-24	Las Vegas (Nevada)	[MedD, MedK]
1534549516631041254	Richard Ross	Male	1975-10-19	San Francisco (California)	[MedE]

Table 2
Anonymized database $\overline{D}_i$ published by S_i.

Label	Name	Gender	Date of Birth	Domicile	Products
516702555509767784	*	Male	$1933 \leq$ Year ≤ 1943	Texas	[MedC]
1807256804637968330	*	Male	$1963 \leq$ Year ≤ 1983	California	[MedA, MedB]
460853062988418469	*	Female	$1993 \leq$ Year ≤ 2008	Nevada	[MedC, MedD]
79983861162328468	*	Male	$1963 \leq$ Year ≤ 1983	California	[MedE]
2176216674885739653	*	Female	$1993 \leq$ Year ≤ 2008	Nevada	[MedC]
5541821146178023331	*	Male	$1933 \leq$ Year ≤ 1943	Texas	[MedL]
3745388544143800788	*	Female	$1993 \leq$ Year ≤ 2008	Nevada	[MedD, MedK]
1534549516631041254	*	Male	$1963 \leq$ Year ≤ 1983	California	[MedE]

Table 3
Database D_k collected by S_k.

Label	Credit card number	Date of Birth	Domicile	Products
7187588859875158153	4254-6266-9975-0706	1933-07-08	Austin(Texas)	[ProdA]
4471466697079625256	4450-5304-6214-5668	1964-11-04	Los Angeles (California)	[ProdA, PodB, ProdC]
7645029893068837442	4821-9429-7881-7361	1993-09-12	Henderson (Nevada)	[ProdA, PodB, ProdD]
1963991313775760113	4223-3060-9605-4063	1966-06-14	San Diego (California)	[ProdE]
4828764993556123852	4667-4851-1088-1447	2004-01-30	Las Vegas (Nevada)	[ProdA, PodD]
2669911912919586508	4842-2302-2803-9399	1934-04-06	Houston (Texas)	[ProdA]
4927142967052839885	4450-5304-6214-5668	1964-11-04	Los Angeles (California)	[ProdB]
3585546642747141943	4355-0290-9842-5202	1975-10-19	San Francisco (California)	[ProdE]

he purchased the product PodB. We suppose the labels associated with these interactions are generated by an analyst in $\mathcal{A}_k$, by using X_k =`AnewSecretAnalyst` as secret.

Similar to S_i, also S_k publishes the anonymized database $\overline{D}_k$ after applying the k-anonymity technique. The resulting database is represented in Table 4.

All the values of the attribute and `Credit Card Number` are suppressed since they are identifiers. As before, the exact values of the attributes `Date of Birth` and `Domicile` are generalized with broader values. The other values of the other attributes are unaltered. Again, we obtain 2-anonymity, so that there are always two entries with the same values of `Date of Birth` and `Domicile`.

Table 4
Anonymized database $\overline{D}_k$ published by S_k.

Label	Credit card number	Date of Birth	Domicile	Products
7187588859875158153	*	$1933 \le$ Year ≤ 1943	Texas	[ProdA]
4471466697079625256	*	$1963 \le$ Year ≤ 1983	California	[ProdA, PodB, ProdC]
7645029893068837442	*	$1993 \le$ Year ≤ 2008	Nevada	[ProdA, PodB, ProdD]
1963991313775760113	*	$1963 \le$ Year ≤ 1983	California	[ProdE]
4828764993556123852	*	$1993 \le$ Year ≤ 2008	Nevada	[ProdA, PodD]
2669911912919586508	*	$1933 \le$ Year ≤ 1943	Texas	[ProdA]
4927142967052839885	*	$1963 \le$ Year ≤ 1983	California	[ProdB]
3585546642747141943	*	$1963 \le$ Year ≤ 1983	California	[ProdE]

```
SecureRandom sr = null;
try
{
sr = SecureRandom.getInstance
("SHA1PRNG");
}
catch (NoSuchAlgorithmException e)
{
}
sr.setSeed(T);
long PrngN = 0;
for (int i=0;i<N;i++)
PrngN=sr.nextLong();
```

Fig. 4. Fragment of code to compute $PRNG(T_i^j, N^j)$ in the analyst module.

```
{
"@context": "http://schema.org/",
"type": "Order",
"customer": {
  "type": "Person",
  "address": {
    "addressLocality": "California"
  },
  "alternateName": "1807256804637968330",
  "birthDate": [
     "1963-01-01",
     "1983-12-31"
    ],
    "gender": "Male"
  },
"orderedItem": [
     {
      "type": "Product",
      "name": "medA"
    },
    {
      "type": "Product",
      "name": "medB"
    }
  ]
}
```

Fig. 5. Anonymized entry of the database $\overline{D}_i$.

5.2. Open data linkage

Through this section, we examine how the analyst A can link the entries in $\overline{D}_i$ and $\overline{D}_k$.

First, A maintains two hash tables H_i and H_k for S_i and S_k, respectively. They are represented in Tables 5 and 6. In Table 5 (Table 6, respectively) the Y associated with a user is mapped to a list of numbers. Each number N included in the list represents the fact that the Nth interaction of the user is performed with S_i (S_k, respectively).

For example, in Table 5, the value [9] (third row) represents the fact that a given user performs the 9th interaction with S_i. The same user (third row in Table 6) performs the 22th interaction with S_k. The other interactions made by the same user, different from the 9th and the 22th, are performed with other service providers not considered in this case study.

Observe that the second entry of both H_i and H_k contains the value Y^j related to John. Specifically in H_i, Y^j is mapped to the value [1], meaning that the first interaction is performed with S_i. While in H_k, Y^j is mapped to the value [2,3], meaning that the second and the third interactions are performed with S_k.

In the following, we describe the steps to perform the linkage. We start from the hash table H_i (related to S_i). For each Y^* in H_i, A computes T_i^* as $MAC(Y^*, X_i)$, where X_i is the secret associated with S_i (in our example X_i is AnalystSecret).

At this point, A retrieves the list L_i^* associated with Y^*. For each N^* in L_i^*, A computes $\hat{P} = PRNG(T_i^*, N^*)$. Then A looks for the entry in $\overline{D}_i$ having as a label the value $\hat{P}$ and replaces it with Y^*.

For example, considering the third entry in H_i, Y^* =eacc4d578a71df946386593e8fcc9a1a5ff5cbecf9d6584a51415fabc8a37803 is associated with $N^* = 9$.

A computes T_i^*, resulting in 7bbbe6dbe0535f876eb19bf137666d09a0855cbc4d7df2743daae8eea8b02c89.

Then, A computes $\hat{P} = PRNG($ 7bbbe6dbe0535f876eb19bf137666d09a0855cbc4d7df2743daae8eea8b02c89$, 9)$.

The result is 460853062988418469, which corresponds to the label of the third entry in $\overline{D}_i$. Then, this label is replaced with Y^* =eacc4d578a71df946386593e8fcc9a1a5ff5cbecf9d6584a51415fabc8a37803.

The result of the above computations is reported in Table 7. For graphical reasons, we report just the first digits of the labels.

The same procedure is performed with the hash table H_k and the anonymized database $\overline{D}_k$. The result is reported in Table 8. Observe that in Table 8, the analyst can already link the second and the seventh row representing two orders made by the same user (in this case John) with S_k.

Finally, A can link the two databases by joining them through the label. The result is reported in Table 9.

6. Security analysis

Through this section, we provide a security analysis of the proposed solution. We start with two basic assumptions.

A1: The used cryptographic functions are secure.
A2: The SSO authentication is secure and prevents impersonation attacks.

Table 5
Hash table H_i.

Y	L
18b9ee4e905baf5c42f342ed5fe03397891910099100f1ec323161b872bbc497	[5]
d94fe9ff76414b9e742819635f7dccf5fddd03c45e201ab34976f2cd9b4459a7	[1]
eacc4d578a71df946386593e8fcc9a1a5ff5cbecf9d6584a51415fabc8a37803	[9]
45f35631d6f5c432a26d31961835ad704e5e4a7934aef090dc5ddab35c027c09	[5]
55d96357d587e955849898d589bce409743cb5efdd3e215c8c37cec1a1b591da	[11]
0e26dcd1d35603ed3ad8c41678e73ee101bbc1029d1a4124973e375347e66fb8	[24]
70e634e66d3388ee23bb8fbc0a4a0538751bb55bb77f2c6178bc47bab97b2e5d	[23]
acd1e2f9e3240103823fed606b6ce8da065660b7e24cc0fab14f9dae192859c6	[31]

Table 6
Hash table H_k.

Y	L
18b9ee4e905baf5c42f342ed5fe03397891910099100f1ec323161b872bbc497	[8]
d94fe9ff76414b9e742819635f7dccf5fddd03c45e201ab34976f2cd9b4459a7	[2,3]
eacc4d578a71df946386593e8fcc9a1a5ff5cbecf9d6584a51415fabc8a37803	[22]
45f35631d6f5c432a26d31961835ad704e5e4a7934aef090dc5ddab35c027c09	[18]
55d96357d587e955849898d589bce409743cb5efdd3e215c8c37cec1a1b591da	[5]
0e26dcd1d35603ed3ad8c41678e73ee101bbc1029d1a4124973e375347e66fb8	[1]
acd1e2f9e3240103823fed606b6ce8da065660b7e24cc0fab14f9dae192859c6	[34]

Table 7
Anonymized database $\hat{D}_i$ after label replacing.

Label	Name	Gender	Date of Birth	Domicile	Products
18b9ee4e...	*	Male	$1933 \leq$ Year ≤ 1943	Texas	[MedC]
d94fe9ff...	*	Male	$1963 \leq$ Year ≤ 1983	California	[MedA, MedB]
eacc4d57...	*	Female	$1993 \leq$ Year ≤ 2008	Nevada	[MedC, MedD]
45f35631...	*	Male	$1963 \leq$ Year ≤ 1983	California	[MedE]
55d96357...	*	Female	$1993 \leq$ Year ≤ 2008	Nevada	[MedC]
0e26dcd1...	*	Male	$1933 \leq$ Year ≤ 1943	Texas	[MedL]
70e634e6...	*	Female	$1993 \leq$ Year ≤ 2008	Nevada	[MedD, MedK]
acd1e2f9...	*	Male	$1963 \leq$ Year ≤ 1983	California	[MedE]

Table 8
Anonymized database $\hat{D}_k$ after label replacing.

Label	Credit card number	Date of Birth	Domicile	Products
18b9ee4e...	*	$1933 \leq$ Year ≤ 1943	Texas	[ProdA]
d94fe9ff...	*	$1963 \leq$ Year ≤ 1983	California	[ProdA, PodB, ProdC]
eacc4d57...	*	$1993 \leq$ Year ≤ 2008	Nevada	[ProdA, PodB, ProdD]
45f35631...	*	$1963 \leq$ Year ≤ 1983	California	[ProdE]
55d96357...	*	$1993 \leq$ Year ≤ 2008	Nevada	[ProdA, PodD]
0e26dcd1...	*	$1933 \leq$ Year ≤ 1943	Texas	[ProdA]
d94fe9ff...	*	$1963 \leq$ Year ≤ 1983	California	[ProdB]
acd1e2f9...	*	$1963 \leq$ Year ≤ 1983	California	[ProdE]

Table 9
Joining of $\hat{D}_i$ and $\hat{D}_k$.

Label	Gender	Date of Birth	Domicile	Products S_i	Products S_k
18b9ee4e...	Male	$1933 \leq$ Year ≤ 1943	Texas	[MedC]	[ProdA]
d94fe9ff...	Male	$1963 \leq$ Year ≤ 1983	California	[MedA, MedB]	[ProdA, ProdB, ProdC],[ProdB]
eacc4d57...	Female	$1993 \leq$ Year ≤ 2008	Nevada	[MedC, MedD]	[ProdA, ProdB, ProdD]
45f35631...	Male	$1963 \leq$ Year ≤ 1983	California	[MedE]	[ProdE]
55d96357...	Female	$1993 \leq$ Year ≤ 2008	Nevada	[MedC]	[ProdA, ProdD]
0e26dcd1...	Male	$1933 \leq$ Year ≤ 1943	Texas	[MedL]	[ProdA]
70e634e6...	Female	$1993 \leq$ Year ≤ 2008	Nevada	[MedD, MedK]	–
acd1e2f9...	Male	$1963 \leq$ Year ≤ 1983	California	[MedE]	[ProdE]

In our setting, **A1** involves the functions MAC and $PRNG$.

This assumption is easily satisfied if the identity provider and analysts use the secure implementations available in the literature for such functions. Some examples, used for our implementation in Section 5, are $HMAC$ based on SHA256 for MAC and $CRNG$ offered by the class `SecureRandom` with `SHA1PRNG` as $PRNG$.

Regarding **A2**, it is a standard requirement and it is realistic since it is adopted in several real-life systems.

Consider a user j interacting n times with a service provider S_i and n^* times with a service provider S_k. j will be associated with the entries $\overline{E_i^j}(1) = \langle P_i^j(1), \overline{D_i^j}(1)\rangle, \dots, \overline{E_i^j}(n) = \langle P_i^j(n), \overline{D_i^j}(n)\rangle$ published by S_i in the database $\overline{D_i}$. Similarly, j will be associated with the entries $\overline{E_k^j}(1) = \langle P_k^j(1), \overline{D_k^j}(1)\rangle, \dots, \overline{E_k^j}(n^*) = \langle P_k^j(n^*), \overline{D_k^j}(n^*)\rangle$ published by S_k in the database $\overline{D_k}$.

We recall that $\mathcal{A}_i$ represents the set of analysts authorized to link the entries published by S_i. Similarly, $\mathcal{A}_k$ represents the set of analysts authorized to link the entries published by S_k.

Our system offers the following properties.

P1: No entity except for S_i and any analyst $A \in \mathcal{A}_i$ can link any pair of labels among $P_i^j(1), \dots, P_i^j(n)$.

P2: No entity, except for any $A \in \mathcal{A}_i \cap \mathcal{A}_k$, can link $P_i^j(1), \dots, P_i^j(n)$ with any among $P_k^j(1), \dots, P_k^j(n^*)$.

P3: Any analyst $A \in \mathcal{A}_i$ can link $P_i^j(1), \dots, P_i^j(n)$ among them.

P4: Any analyst $A \in \mathcal{A}_i \cap \mathcal{A}_k$ can link $P_i^j(1), \dots, P_i^j(n)$ with any among $P_k^j(1), \dots, P_k^j(n^*)$.

P5: A user $\hat{j}$ cannot make S_i publish any entry with the label $P_i^{\hat{j}}(z)$ (associated with the zth interaction of $\hat{j}$ with S_i) linkable with any among $P_i^j(1), \dots, P_i^j(n)$ and $P_k^j(1), \dots, P_k^j(n)$.

Observe that the first four properties reflect the two requirements introduced at the end of Section 3. Instead, the property **P5** concerns the problem of impersonation attacks. Even though it is not the main focus of this proposal, our solution addresses it.

Property P1

Consider a pair $P_i^j(x)$ and $P_i^j(y)$.

Obviously, S_i knows the label belonging to each user, since each label is built after the interaction with a user. Then, in the following, we consider an attacker different from S_i.

We recall that $P_i^j(x)$ is obtained as $PRNG(T_i^j, \overline{N^j})$ for some value of $\overline{N^j}$. Similarly, $P_i^j(y)$ is obtained as $PRNG(T_i^j, \widehat{N^j})$ for some value of $\widehat{N^j}$. We recall that T_i^j is uniquely associated with j (and S_i).

To link $P_i^j(x)$ and $P_i^j(y)$, the attacker should know the pairs $\langle T_i^j, \overline{N^j}\rangle$ and $\langle T_i^j, \widehat{N^j}\rangle$. The values $\overline{N^j}$ and $\widehat{N^j}$ can be easily guessed by brute force. On the other hand, T_i^j cannot be retrieved by any entity different from an analyst in $\mathcal{A}_i$. Indeed, by Assumption **A1**, the PRNG cannot be reversed.

Therefore, no entity different from an analyst in $\mathcal{A}_i$ and S_i can link $P_i^j(x)$ with $P_i^j(y)$.

Property P2

We follow the same reasoning of Property **P1**. Consider a pair $P_i^j(x)$ and $P_k^j(y)$, where $P_i^j(x)$ is obtained as $PRNG(T_i^j, \overline{N^j})$ for some value of $\overline{N^j}$, and $P_k^j(y)$ is obtained as $PRNG(T_k^j, \widehat{N^j})$ for some value of $\widehat{N^j}$.

To link $P_i^j(x)$ and $P_k^j(y)$, the attacker should know the pairs $\langle T_i^j, \overline{N^j}\rangle$ and $\langle T_k^j, \widehat{N^j}\rangle$. The values $\overline{N^j}$ and $\widehat{N^j}$ can be easily guessed by brute force. On the other hand, by Assumption **A1**, the PRNG cannot be reversed and then T_i^j and T_k^j cannot be retrieved by any entity different from an analyst in $\mathcal{A}_i \cap \mathcal{A}_k$. Indeed, the analysts in $\mathcal{A}_i \backslash \mathcal{A}_k$ only know T_i^j and the analysts in $\mathcal{A}_k \backslash \mathcal{A}_i$ only know T_k^j.

Therefore, no entity different from an analyst in $\mathcal{A}_i \cap \mathcal{A}_k$ can link $P_i^j(x)$ with $P_k^j(y)$.

Property P3

Consider two labels $P_i^j(x)$ and $P_i^j(y)$ assigned during the xth and yth interaction of j with S_i, respectively.

We recall that $P_i^j(x)$ is obtained as $PRNG(T_i^j, \overline{N^j})$ for some value of $\overline{N^j}$, and $P_i^j(y)$ is obtained as $PRNG(T_i^j, \widehat{N^j})$ for some value of $\widehat{N^j}$.

Then, the analyst A can link $P_i^j(x)$ and $P_i^j(y)$ if it knows $T_i^j, \overline{N^j}, \widehat{N^j}$. The values $\overline{N^j}$ and $\widehat{N^j}$ are provided to A by S_i (along with a value Y^j) during the xth and yth interaction, respectively, of j with S_i. These values are stored locally by A and associated with Y^j. Observe that, since Y^j is computed by the identity provider as $Y^j = MAC(I^j, Secr^j)$, it is the same for the interactions x and y of j with S_i. Then, the analyst A, once receiving Y^j, can compute $T_i^j = MAC(Y^j, X_i)$ since it knows the secret X_i and link $P_i^j(x)$ with $P_i^j(y)$.

Property P4

The reasoning is similar to Property **P3**. Consider two labels $P_i^j(x)$ and $P_k^j(y)$ assigned during the xth interaction of j with S_i and yth interaction of j with S_k, respectively.

We recall that $P_i^j(x)$ is obtained as $PRNG(T_i^j, \overline{N^j})$ for some value of $\overline{N^j}$, and $P_k^j(y)$ is obtained as $PRNG(T_k^j, \widehat{N^j})$ for some value of $\widehat{N^j}$.

Then, the analyst A can link $P_i^j(x)$ and $P_k^j(y)$ if it knows $T_i^j, T_k^j, \overline{N^j}, \widehat{N^j}$. Indeed, the values $T_i^j = MAC(Y^j, X_i)$ and $T_k^j = MAC(Y^j, X_k)$ are computed starting from the same value Y^j.

Since the analyst A knows both X_i and X_k, it can link T_k^j with T_i^j if it knows Y^j. Y^j is provided during the xth interaction of j (along with $\overline{N^j}$) with S_i and the yth interaction of j with S_k (along with $\widehat{N^j}$).

Once linking T_k^j with T_i^j, A can compute $P_i^j(x)$ and $P_k^j(y)$, and link them.

Property P5

This property can be broken when the label $P_i^{\hat{j}}(z)$ is linkable with a label $P_i^j(x)$ (obtained by S_i during the xth interaction with j) or $P_k^j(y)$ (obtained by S_k during the yth interaction with j).

We recall that $P_i^{\hat{j}}(z) = PRNG(T_i^{\hat{j}}, N_1)$, $P_i^j(x) = PRNG(T_i^j, N_2)$, and $P_k^j(y) = PRNG(T_k^j, N_3)$ for some N_1, N_2, N_3. Moreover, we have that $T_i^{\hat{j}} = MAC(Y^{\hat{j}}, X_i)$, $T_i^j = MAC(Y^j, X_i)$, and $T_k^j = MAC(Y^j, X_k)$.

Then, we have $P_i^{\hat{j}}(z)$ is linkable to $P_i^j(x)$ or to $P_k^j(y)$, if $Y^j = Y^{\hat{j}}$.

Since $Y^j = MAC(I^j, Secr^j)$, by Assumption **A1** (no collision of the hash function), $Y^j = Y^{\hat{j}}$ occurs only if $I^j = I^{\hat{j}}$ and $Secr^j = Secr^{\hat{j}}$.

Since these values are stored by the identity provider IP, this case occurs only if $\hat{j}$ authenticates with IP in place of j. This cannot occur by Assumption **A2** (impersonation attacks are not possible).

7. Related work

Despite all the benefits coming from the exploitation of open data in different scenarios, many privacy issues may arise when dealing with data about individual preferences and behaviors [30]. As a matter of fact, removing all the obviously identifiable information from a given dataset is not enough to prevent individual re-identification.

Traditional solutions to protect individual privacy (thus preventing the above-mentioned attack) are based on the notions of k-anonymity [10], l-diversity [11] and t-closeness [12]. Unfortunately, these methods may still leak information when the attackers already know something (background knowledge) about the information contained in the dataset [31].

More advanced solutions to protect individual privacy are based on differential privacy [32], which is considered among the most promising paradigms for privacy-preserving data publication and analysis [33]. Many approaches, employing differential privacy, are based on adding noise to the data before disclosing them [13]. Nevertheless, a drawback of differential privacy is that the presence of noise may lead to a low utility of the released data [34].

An emerging technique to obtain differential privacy, involves the use of Generative Adversarial Networks (GANs) [35–37]. Such a technique is used by Frigerio et al. [38], who propose a framework for releasing new open data while protecting user privacy.

When dealing with open data, a challenging issue is represented by the lack of links between data when the above-mentioned privacy approaches are adopted by different sources (possibly using different anonymizing techniques). On the other hand, linking data related to the same individual, but distributed among different datasets, allows for more powerful and efficient analysis [39].

Concerning data linkage, [39,40] provide an overview of privacy issues related to linkable data. However, they do not propose any solution to address this problem.

As highlighted by [41,42], performing the linkage process would intrinsically require the presence of a common unique identifier for all the data belonging to the same user.

Another problem concerning open data is the fact that multiple organizations may independently release anonymized open data about overlapping populations. Indeed, an attacker may break individuals' privacy by linking such data among them. This attack is known as *composition attack* [14,15].

In principle, such an attack is also possible when a data owner sequentially releases anonymized datasets over time [43–46]. However, unlike the previous case, since all the datasets are published by the same data owner, it can use the information in the previously published datasets to anonymize the current dataset and thus counter composition attacks [14]. Conversely, when the datasets are published by independent sources, different solutions must be employed to counter the above attack.

Randomization-based techniques, such as differential privacy, have been proven to be effective in countering composition attacks [15]. However, as highlighted above, these techniques may lead to a low utility of the data released due to the presence of noise. Clearly, these techniques alone do not only prevent composition attacks but also the linkage among data. Therefore, additional solutions must be employed to enable such linkage.

A different approach to counter composition attacks consists in adopting a distributed model that allows multiple data owners to collaborate with each other to properly anonymize their datasets before publishing them [47]. Often this approach leverages Secure Multiparty Computation (SMC) techniques [48,49]. These techniques allow multiple data owners to perform a joint computation of an anonymized dataset, while preventing each owner from sharing its original dataset with the other parties.

For instance, [50,51] leverage SMC to enable multiple parties to generate k-anonymous datasets without revealing their data to each other. Similarly, Goryczka et al. [52] address the collaborative data publishing problem by taking into account colluding data owners that may use their own data records to infer the data records contributed by other data owners.

A similar issue is addressed in [53]. However, unlike the above-presented solutions, Mohammed et al. [53] present a collaborative solution that does not leverages SMC. Indeed SMC allows sharing the final result while it prohibits sharing the input of the computation. On the contrary, the solution proposed in [53] allows the disclosure of local data that satisfy a given k-anonymity requirement.

Observe that the above-mentioned solutions require the interaction among data owners aimed to jointly publish an anonymized version of the linked data. On the contrary, our solution does not require any interaction among the service providers, which would be, in the context of open data, little realistic. Moreover, the above-mentioned solutions cannot be directly applied to our context since they would publicly disclose the linkage among different datasets, while our goal is to allow only authorized parties to learn this information.

8. Conclusion

In this paper, we propose a solution for the linkage of open data published by different sources. The advantage of our solution is that only some authorized parties can perform this linkage. This enables more efficient analyses and prevents unnecessary privacy leakage with respect to non-authorized parties. The proposed solution is shown to be concretely applicable by implementing it in the SAML-based SSO authentication framework compliant with the eIDAS regulation.

An aspect that has not been investigated in-depth in this paper regards the anonymizing function δ. Indeed, even though our α function does not introduce any privacy leakage with respect to any unauthorized entity, it is not clear if the linkage of the published open data, by an authorized entity, may reveal further sensitive information about the user (such as their identity) beyond the linkage itself (for example, through composition attacks on anonymized databases [15]). Actually, it depends on the data, the background knowledge of the adversary, and the anonymized function used. As future work, we plan to understand whether our solution is compatible with advanced privacy-preserving techniques (such as [14,15] resistant to composition attacks) that mitigate privacy leakage when the linkage of anonymized data is enabled.

CRediT authorship contribution statement

Francesco Buccafurri: Conceptualization, Methodology, Formal analysis, investigation, validation, Writing – original draft, Writing – review & editing, Supervision, Project administrator. **Vincenzo De Angelis:** Conceptualization, Methodology, Formal analysis, investigation, validation, Writing – original draft, Writing – review & editing, Software, Resources, Data curation, Visualization. **Sara Lazzaro:** Conceptualization, Methodology, Formal analysis, Investigation, Validation, Writing – original draft, Writing – review & editing, Software, Resources, Data curation, Visualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article

Acknowledgments

This work was partially supported by the project STRIDE included in the Spoke 5 (Cryptography and Distributed Systems Security) of the Research and Innovation Program PE00000014, “SEcurity and RIghts in the CyberSpace (SERICS)”, under the National Recovery and Resilience Plan, funded by the European Union, NextGenerationEU.

References

[1] Hopfgartner F, Jose JM. Semantic user profiling techniques for personalised multimedia recommendation. Multimedia Syst 2010;16:255–74.
[2] Murray-Rust P. Open data in science. Nat Proc 2008;1.
[3] Kitchin R. The data revolution: Big data, open data, data infrastructures and their consequences. 2014.
[4] Wilson B, Cong C. Beyond the supply side: Use and impact of municipal open data in the us. Telemat Inform 2021;58:101526.
[5] Begany GM, Gil-Garcia JR. Understanding the actual use of open data: Levels of engagement and how they are related. Telemat Inform 2021;63:101673.
[6] Zuiderwijk A, Janssen M, Poulis K, van de Kaa G. Open data for competitive advantage: insights from open data use by companies. In: Proceedings of the 16th annual international conference on digital government research. 2015, p. 79–88.
[7] Daries JP, Reich J, Waldo J, Young EM, Whittinghill J, Ho AD, Seaton DT, Chuang I. Privacy, anonymity, and big data in the social sciences. Commun ACM 2014;57:56–63.
[8] Ni C, Cang LS, Gope P, Min G. Data anonymization evaluation for big data and iot environment. Inform Sci 2022;605:381–92.
[9] Varanda A, Santos L, Costa RLdC, Oliveira A, Rabadão C. Log pseudonymization: Privacy maintenance in practice. J. Inf Secur Appl 2021;63:103021.
[10] Samarati P, Sweeney L. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. 1998.
[11] Machanavajjhala A, Kifer D, Gehrke J, Venkitasubramaniam M. L-diversity: Privacy beyond k-anonymity. ACM Trans Knowl Discov Data (TKDD) 2007;1:3–es.
[12] Li N, Li T, Venkatasubramanian S. T-closeness: Privacy beyond k-anonymity and l-diversity. In: 2007 IEEE 23rd international conference on data engineering. IEEE; 2007, p. 106–15.
[13] Dwork C, Kenthapadi K, McSherry F, Mironov I, Naor M. Our data, ourselves: Privacy via distributed noise generation. In: Annual international conference on the theory and applications of cryptographic techniques. Springer; 2006, p. 486–503.
[14] Li J, Baig MM, Sattar AS, Ding X, Liu J, Vincent MW. A hybrid approach to prevent composition attacks for independent data releases. Inform Sci 2016;367:324–36.
[15] Ganta SR, Kasiviswanathan SP, Smith A. Composition attacks and auxiliary information in data privacy. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining. 2008, p. 265–73.
[16] Union E. Regulation EU no 910/2014 of the European parliament and of the council. 2014, http://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX%3A32014R0910&from=EN (Last checked 21/09/2022).

[17] Radha V, Reddy DH. A survey on single sign-on techniques. Proc Technol 2012;4:134–9.
[18] Hughes J, Maler E. Security assertion markup language (saml) v2. 0 technical overview. OASIS SSTC working draft sstc-saml-tech-overview-2.0-draft-08 13, 2005.
[19] Berners-Lee T. Star deployment scheme for open data. 2016, http://5stardata.info/ Accessed on 2022 10 (5).
[20] Bauer F, Kaltenböck M. Linked open data: the essentials, Vol. 710. Vienna; 2011, Edition mono/monochrom.
[21] Sikos LF, Philp D. Provenance-aware knowledge representation: A survey of data models and contextualized knowledge graphs. Data Sci Eng 2020;5:293–316.
[22] Sakimura N, Bradley J, Jones M, De Medeiros B, Mortimore C. Openid connect core 1.0. The OpenID Foundation; 2014, p. S3.
[23] Sweeney L. K-anonymity: A model for protecting privacy. Int J Uncertain Fuzziness Knowl-Based Syst 2002;10:557–70.
[24] Christie MA, Bhandar A, Nakandala S, Marru S, Abeysinghe E, Pamidighantam S, Pierce ME. Using keycloak for gateway authentication and authorization. 2017.
[25] Perry BW. Java servlet & jsp cookbook. 2004.
[26] Sporny M, Longley D, Kellogg G, Lanthaler M, Lindström N. Json-ld 1.0. 2014, p. 41, W3C recommendation 16.
[27] Krawczyk H, Bellare M, Canetti R. Hmac: Keyed-hashing for message authentication. 1997.
[28] Özkaynak F. Cryptographically secure random number generator with chaotic additional input. Nonlinear Dynam 2014;78:2015–20.
[29] Group WSC. Schema.org project. 2022, https://github.com/schemaorg/schemaorg (Last checked 21/09/2022).
[30] Jaatinen T. The relationship between open data initiatives, privacy, and government transparency: a love triangle? Int Data Priv Law 2016;6:28.
[31] Ji Z, Lipton ZC, Elkan C. Differential privacy and machine learning: a survey and review. 2014, arXiv preprint arXiv:1412.7584.
[32] Dwork C. Differential privacy: A survey of results. In: International conference on theory and applications of models of computation. Springer; 2008, p. 1–19.
[33] Yang Y, Zhang Z, Miklau G, Winslett M, Xiao X. Differential privacy in data publication and analysis. In: Proceedings of the 2012 ACM SIGMOD international conference on management of data. 2012, p. 601–6.
[34] Mohammed N, Chen R, Fung BC, Yu PS. Differentially private data release for data mining. In: Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining. 2011, p. 493–501.
[35] Xu C, Ren J, Zhang D, Zhang Y, Qin Z, Ren K. Ganobfuscator: Mitigating information leakage under gan via differential privacy. IEEE Trans Inf Forensics Secur 2019;14:2358–71.
[36] Zhang X, Ji S, Wang T. Differentially private releasing via deep generative model. Technical report, 2018, arXiv preprint arXiv:1801.01594.
[37] Xie L, Lin K, Wang S, Wang F, Zhou J. Differentially private generative adversarial network. 2018, arXiv preprint arXiv:1802.06739.
[38] Frigerio L, d. Oliveira AS, Gomez L, Duverger P. Differentially private generative adversarial networks for time series, continuous, and discrete open data. In: IFIP international conference on ICT systems security and privacy protection. Springer; 2019, p. 151–64.
[39] Harron K, Dibben C, Boyd J, Hjern A, Azimaee M, Barreto ML, Goldstein H. Challenges in administrative data linkage for research. Big Data Soc 2017;4:2053951717745678.
[40] Zheng X, Cai Z, Li Y. Data linkage in smart internet of things systems: a consideration from a privacy perspective. IEEE Commun Mag 2018;56:55–61.
[41] Christen P, Churches T, Hegland M. Febrl–a parallel open source data linkage system. In: Pacific-Asia conference on knowledge discovery and data mining. Springer; 2004, p. 638–47.
[42] Smith D. Secure pseudonymisation for privacy-preserving probabilistic record linkage. J Inf Secur Appl 2017;34:271–9.
[43] Fung BC, Wang K, Fu AW-C, Pei J. Anonymity for continuous data publishing. In: Proceedings of the 11th international conference on extending database technology: advances in database technology. 2008, p. 264–75.
[44] Xiao X, Tao Y. M-invariance: towards privacy preserving re-publication of dynamic datasets. In: Proceedings of the 2007 ACM SIGMOD international conference on management of data. 2007, p. 689–700.
[45] He Y, Barman S, Naughton JF. Preventing equivalence attacks in updated, anonymized data. In: 2011 IEEE 27th international conference on data engineering. IEEE; 2011, p. 529–40.
[46] Wong RC-W, Fu AW-C, Liu J, Wang K, Xu Y. Global privacy guarantee in serial data publishing. In: 2010 IEEE 26th international conference on data engineering (ICDE 2010). IEEE; 2010, p. 956–9.
[47] Fung BC, Wang K, Chen R, Yu PS. Privacy-preserving data publishing: A survey of recent developments. ACM Comput Surv (Csur) 2010;42:1–53.
[48] Yao AC. Protocols for secure computations. In: 23rd annual symposium on foundations of computer science (Sfcs 1982). IEEE; 1982, p. 160–4.
[49] Yao AC-C. How to generate and exchange secrets. In: 27th annual symposium on foundations of computer science (Sfcs 1986). IEEE; 1986, p. 162–7.
[50] Jiang W, Clifton C. A secure distributed framework for achieving k-anonymity. VLDB J 2006;15:316–33.
[51] Jurczyk P, Xiong L. Privacy-preserving data publishing for horizontally partitioned databases. In: Proceedings of the 17th ACM conference on information and knowledge management. 2008, p. 1321–2.
[52] Goryczka S, Xiong L, Fung BC. *m*-Privacy for collaborative data publishing. IEEE Trans Knowl Data Eng 2013;26:2520–33.
[53] Mohammed N, Fung BC, Wang K, Hung PC. Privacy-preserving data mashup. In: Proceedings of the 12th international conference on extending database technology: advances in database technology. 2009, p. 228–39.