Towards Privacy-preserving Attribute Aggregation in Federated eID Systems

Walter Priesnitz Filho[1], Carlos Ribeiro[1], and Thomas Zefferer[2]

Universidade de Lisboa - Instituto Superior Técnico
Rovisco Pais 1, Lisboa, Portugal,
{walter.filho | carlos.ribeiro}@tecnico.ulisboa.pt,
Graz University of Technology - Institute for Applied Information Processing and Communications, Inffeldgasse 16a, A-8010 Graz, Austria
thomas.zefferer@iaik.tugraz.at

Abstract. During the past years, achieving interoperability, i.e. creating identity federations, between different eID systems has gained relevance. A key problem of identity federations is the missing harmonization between different attribute providers (APs). In closed eID systems, ontologies allow a higher degree of automation in the process of aligning and aggregating attributes from different APs. This approach does not work for identity federations, as each eID system uses its own ontology to represent attributes. Moreover, providing attributes to intermediate entities required to align and aggregate attributes potentially violates privacy rules. To tackle these problems, we propose the use of combined ontology alignment approaches and locality-sensitive hashing (LSH) functions. We assess existing implementations of these concepts by means of criteria that are specific for identity federations. Obtained results show that suitable implementations of these concepts exist and that they can be used to achieve interoperability between eID systems on attribute level.

Keywords: interoperability, ontologies, LSH Functions, privacy.

1 Introduction

Electronic identities (eID) have become a crucial concept of electronic services from both the private and the public sector. For instance, e-government solutions use eIDs to identify and authenticate citizens in governmental online processes.

An eID process involves several entities. The Identity Provider (IdP) establishes and verifies the identity of the user. The Relying Party (RP) makes transaction decisions based on receipt and validation of a user's authenticated credentials within the Identity System (IS). For instance, a Service Provider (SP) can assume the role of the RP.

In most ISs, identity attributes are used together with the eID of a user. For instance, the RP might additionally be provided with the user's name and date of birth. From a conceptual perspective, identity attributes are provided by

In: S. España, M. Ivanović, M. Savić (eds.): Proceedings of the CAiSE'16 Forum at the 28th International Conference on Advanced Information Systems Engineering, Ljubljana, Slovenia, 13-17.6.2016, published at http://ceur-ws.org

Attribute Providers (APs). In practice, IdP and AP can also be represented by one and the same entity.

Several attempts have been made recently to achieve interoperability between ISs, i.e. to establish an eID federation. In Europe, the EU large scale pilots STORK[1] and STORK 2.0[2] have successfully established eID interoperability between EU member states (MS). As a result, citizens from MS A can use their national eID to identify and authenticate at SPs from MS B and vice versa.

Note that in most cases interoperability of eID attributes is implicitly assumed. In scenarios that involve multiple ISs this assumption is usually not valid and attributes cannot be easily exchanged between entities. In this paper, we address this issue. We propose the use of ontology-alignment (OA) solutions and locality-sensitive hashing (LSH) functions to aggregate attributes in eID federations. We survey existing implementations of these two technologies and assess them. This way, this paper represents a significant step towards privacy-preserving attribute aggregation in federated eID systems.

2 Survey

Figure 1 shows the problem addressed in this paper. An RP requests attributes from two APs of different ISs. To cope with the different ontologies of these ISs, an Aggregation Entity (AE) is employed, which acts as gateway. To achieve attribute interoperability, the AE employs two technologies, i.e. OA solutions and LSH functions.

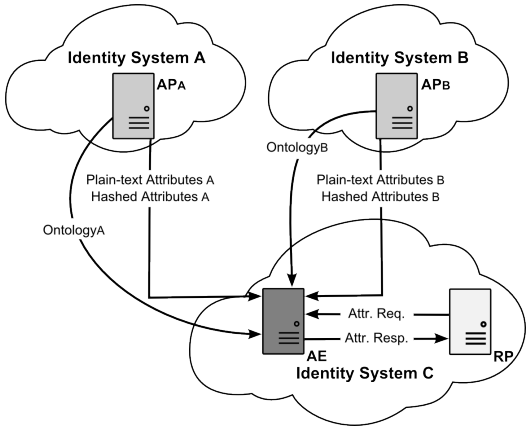

Fig. 1. General architecture.

[1] https://www.eid-stork.eu/
[2] https://www.eid-stork2.eu/

Ontologies are a useful concept to facilitate the use of attributes from different APs. To cope with different ontologies from different ISs, the AE can follow an OA approach. This enables the AE to create an alignment describing the relation between different ontologies.

To achieve a reliable OA, the AE potentially needs to request more attributes from APs than originally requested by the RP. Unfortunately, this can violate minimum-disclosure rules and hence reduce privacy. To address this issue, we propose the use of LSH functions. In contrast to ordinary hash functions, LSH functions reflect the similarity of input values to derived hash values. This way, relevant information for OA purposes can be exchanged without violating privacy rules.

For both technologies employed, various implementation already exist. In the following subsections, we survey these implementations to provide a solid foundation for subsequent assessments.

2.1 Existing Ontology-Alignment Solutions

The OA solutions are used to obtain a common knowledge representation among entities. Two or more ontologies are aligned to enable involved entities to use a common vocabulary to communicate with each other. In the following, three of the most commonly used solutions that can be applied in OA are briefly sketched.

- **AlignAPI:** The Alignment API (AlignAPI[3]), available in Java, can be used for representing OAs and for the development, integration and composition of matchers. It provides examples and the basic tools for manipulating OAs [1]. Its reference implementation enables development of tools for manipulating OAs and calling matchers. The AlignAPI is a basic tool that helps matcher developers to deliver OAs in a well-supported framework [2].
- **PROMPT:** PROMPT[4] is an algorithm and a tool for merging and aligning ontologies [3]. It requires direct interaction with the user. The tool takes two ontologies as input [4] and guides the user through the process. Initially, PROMPT creates a list of matches considering class names. Then, it carries out the following steps in a loop: (1) The user selects one of the suggestions or edits the ontology. (2) PROMPT performs the operation, makes necessary changes, generates a list of suggestions for the user, determines conflicts generated, and finds solutions for those conflicts.
- **XMAP:** The XMAP[5] ontology-matching system is able to perform matching on large ontologies [5]. A semantic similarity measure is defined using UMLS and WordNet[6] to provide a synonymy degree between two entities from different ontologies by exploring both their lexical and structural context. XMAP exploits the common elements from the descriptions to measure the similarity between two classes and two properties, respectively.

[3] http://alignapi.gforge.inria.fr/
[4] http://protegewiki.stanford.edu/wiki/PROMPT
[5] http://www.labged.net/index.php?rubrique=mapage38
[6] https://wordnet.princeton.edu/

2.2 Existing LSH Functions

LSH functions map similar objects into the same hash buckets with high probability. LSH functions ensure that the collision probability is higher for closer objects (objects with similar values) than for farer objects (objects with different values) [6, 7]. In the following, we briefly sketch existing implementations.

- **MinHash:** MinHash techniques evaluate the similarity of any two sets requiring only a constant number of comparisons [8]. MinHash works by extracting a representation $h_k(S)$ of a set S using a deterministic sampling. This $h_k(S)$ has a constant size k, independent from $|S|$. The computation of $h_k(S)$ incurs a complexity linear in set sizes.
- **Nilsimsa:** Nilsimsa [9] is an LSH function that takes an arbitrary input and outputs an n-bit digest. It uses n buckets to count the trigrams that appear in the input and converts the counts to an n-bit digest. The assess the similarity between two inputs [10], the algorithm counts the number of equal bits of the two Nilsimsa digests in the same position.
- **TLSH:** The TLSH value is determined by summing up the distance between the digest headers and the digest bodies. The resulting distance score ranges from 0 to 1000+. Digests with a $distance \leq 100$ are considered to be similar. Digests with a $distance > 100$ are considered as not similar [11].

3 Assessment

To determine the best existing implementations of OA solutions and LSH functions, we conduct assessments on all solutions surveyed. In the following, relevant assessment criteria are derived and obtained assessment results are presented.

3.1 Assessment Criteria

Assessment of the surveyed implementations has been based on a set of assessment criteria. All criteria have been defined such that they enable assessment of the surveyed solutions with regard to their effectiveness and ease of integration.

Some of the criteria defined are relevant for both LSH functions and OA solutions. Most of them are non-technical, but still relevant especially with regard to a solution's ease of integration. This includes the following criteria:

- **Documentation (Doc):** related to the amount and quality of resources that are available;
- **Implementation (Imp):** the programming language (PL) used or interfaces provided; and
- **Processing Time (PT):** describes the time required to execute a given task.

In addition to the criteria that are relevant for both OA solutions and LSH functions, several criteria can be identified that are relevant for LSH functions only. This includes the following criteria:

- **Function Score (F-S) and Clear-Text Score (CT-S):** The F-S and Ct-S provide the function's obtained score (between signatures) and the absolute similarity score (between clear texts) in a given test. The values are computed using the Levenshtein Distance (LD) between the values provided as input and are normalized considering the signature length.
- **Similarity Scale (SSca):** describes SSca used by the evaluated implementation.

Finally, specific assessment criteria can also be defined for OA solutions. They include the following aspects:

- **Similarity Scores (SSco):** provides the values, considering the SSca of the evaluated solution, obtained on the assessment;
- **Mappings Identified (MI):** describes the absolute number of correspondences on both evaluated ontologies.

Based on these criteria, all surveyed implementations have been assessed. Results obtained from these assessments are summarized in the following.

3.2 Assessment Results

Mapping the defined assessment criteria to the implementations surveyed has yielded interesting results. These results are presented in this section.

Assessment Results of Ontology-Alignment Solutions All surveyed OA solutions have diverse sources of documentation. The surveyed implementations are available either as Java API or as Protégé Plugin and have been published under LGPL or MPL. Table 1 summarizes all results concerning available documentation and implementation. We ran some tests to compare the implementations'

Solution	Doc.	Imp.	License
AlignAPI	Web Page, Tutorial	Java API	LGPL
PROMPT	Wiki	Java API, Protégé Plugin	MPL
XMAP	Web Page (last update 2011)	Protégé Plugin	N/A

Table 1. Documentation, Implementation, and License (LGPL - GNU Lesser General Public License or MPL - Mozilla Public License) assessments.

effectiveness and efficiency. Tests were performed by loading two ontologies, O_1 and O_2 (available online as storkPerson[7] and storkPerson1[8]), and by verifying obtained matches. The two ontologies used had the same concepts, but four of

[7] http://web.tecnico.ulisboa.pt/walter.filho/ontologies/STORK/storkPerson.owl
[8] http://web.tecnico.ulisboa.pt/walter.filho/ontologies/STORK/storkPerson1.owl

them had been written with similar terms, namely: eMail ↔ e-mail; dateOfBirth ↔ birthDay; surname ↔ lastName; and givenName ↔ name.

Since the ontologies O_1 and O_2 present previously known similar terms, we were able to verify the accuracy of each OA solution. Results obtained are summarized in Table 2.

Solution	PT	SSco	MI
AlignAPI	1.467s	0.96807	22
PROMPT	4.036s	Not provided	1
XMAP	Not finished	N/A	N/A

Table 2. Process. time, Sim. Scores, and Mappings assessments.

We also investigated the PT on the surveyed solutions. Running the test with the ontologies O_1 and O_2 revealed that the AlignAPI was 2.75x faster than PROMPT.

The tests also showed significant differences between AlignAPI and PROMPT solutions regarding the number of matchings found. Running the tests with O_1 and O_2 yielded 22 matchings found by AlignAPI, and only one matching found by PROMPT. Table 2 illustrates these results.

Summarizing the results obtained, it can be concluded that AlignAPI is the most suitable solution. This solution outperforms other alternatives with regard to performance and ease of integration.

Assessment Results of LSH Functions Similar to the surveyed OA solutions, all surveyed LSH functions provide diverse sources of documentation and are available in several PLs. All surveyed solutions are available under the Apache License (AL). The SSca of each solution was evaluated as well. Obtained results are summarized in Table 3.

Solution	Doc.	Imp.	License	SSca
MinHash	GSCW	Group 1	AL	0 - 10
Nilsimsa	GSCW	Group 1 + Group 2	AL	128 - (-128)
TLSH	GitHub	Python, Java, JS	AL	0 - 200 / 0 - 400

Table 3. Documentation (G - GitHub, S - Sample Code, W - WikiPedia), Implementation (Group 1: Java, Python, Ruby, PHP, Go, C; Group 2: C++, C#), Similarity Scale, and License assessments.

We also ran tests to assess the performance of the different solutions. Note that the TLSH[9] function is unable to produce results on inputs with a size

[9] https://github.com/trendmicro/tlsh

smaller than 256 bytes. This constraint makes this solution unsuitable for use cases related to eID attributes, as they are potentially very short. For this reason, TLSH was precluded from further performance assessments.

Four tests have been defined and run to assess the performance of the remaining two LSH solutions. The tests compute the similarity between the hash values from user names (HN_1 and HN_2), i.e. a typical eID attribute. We generated and used a set of 1,000 records with random data (i.e.: given name, family name, birthday, etc.) to run these tests. For each test, slightly different input values have been used as defined below:

1. **Test 1:** HN_1 and HN_2 received the same full user name (F-UN);
2. **Test 2:** HN_1 received the F-UN and HN_2 received the same F-UN provided but used some abbreviation.
3. **Test 3:** HN_1 and HN_2 received the same set of 1,000 F-UN.
4. **Test 4:** HN_1 received a set of 1,000 F-UN and HN_2 received a set of 1,000 F-UN provided but used some abbreviation.

Execution of Test 1 provided the metrics of each function performing the analysis of just one element (record) with a full match. Test 2 provided a similar metric, but considered similar values. Test 3 used 1,000 records with full matches, a scenario close to a real-world. The same applied to Test 4, which combined the specifics of Test 2 and Test 3. Table 4 shows the results obtained.

Solution	Test 1			Test 2			Test 3			Test 4		
	Time	F-S	Ct-S	Time	F-S	Ct-S	Time	F-S	Ct-S	Time	F-S	Ct-S
MinHash	90.73	0.941	0	94.08	0.941	5	597.4	0.847	0	518.7	0.847	0.288
Nilsimsa	2.074	0	0	3.169	0.56	5	278.5	0	0	327.0	0.667	0.288

Table 4. Tests results for LSH Functions

Regarding the PT, Nilsimsa was from 43x (Test1) to 1.6x (Test 4) faster than MinHash . We used the LD to gauge the similarity of input values and output values. Results showed that Nilsimsa yielded closer values to the LD of input text than the MinHash . For Test 1 and Test 3, the result was exactly the same. For Test 4, the result of Nilsimsa was 25% closer than the outcome of MinHash.

Table 4 summarizes assessment results of the surveyed LSH Functions. From the results obtained, we conclude that Nilsimsa is the winner, as it outperforms other solutions and complies best with relevant assessment criteria.

4 Conclusions and Future Work

In this paper we have proposed the use of ontology alignment and LSH functions to leverage attribute interoperability in eID federations. A survey conducted has revealed that implementations of these two technologies already exist but are

not tailored to the use case given. We have hence applied systematic assessments of available solutions by means of relevant criteria. These assessments have revealed that AlignAPI [1] is the most suitable OA solution available. Furthermore, Nilsimsa [9, 10] turned out to be the LSH implementation that meets best the special requirements of eID federations.

In future work, we will use results obtained to realize an AE as illustrated in Figure 1. This AE will finally enable attribute interoperability in eID federations. The work presented in this paper is a fundamental basis, as it assures that the two most relevant building blocks of the AE, i.e. ontology alignment and LSH functionality, are implemented in the most effective and efficient way.

Acknowledgment. This work was partially supported by CAPES Proc. Num. BEX 9096/13-2 and EU project Stork 2.0 CIP-ICT-PSP-2011-5-297263.

References

1. David, J., Euzenat, J., Scharffe, F., dos Santos, C.T.: The alignment api 4.0. Semantic Web journal **2** (2011) 3–10
2. Euzenat, J.: An api for ontology alignment. In: The Semantic Web-ISWC 2004. Springer (2004) 698–712
3. Malik, S., Prakash, N., Rizvi, S.: Ontology merging using prompt plug-in of protg; in semantic web. In: Computational Intelligence and Communication Networks (CICN), 2010 International Conference on. (Nov 2010) 476–481
4. Noy, N.F., Musen, M.A.: Algorithm and tool for automated ontology merging and alignment. In: Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00). Available as SMI technical report SMI-2000-0831. (2000)
5. Djeddi, W.E., Khadir, M.T., Yahia, S.B.: Xmap: Results for oaei 2015
6. Indyk, P., Motwani, R.: Approximate nearest neighbors: Towards removing the curse of dimensionality. In: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing. STOC '98, New York, NY, USA, ACM (1998) 604–613
7. Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.S.: Locality-sensitive hashing scheme based on p-stable distributions. In: Proceedings of the Twentieth Annual Symposium on Computational Geometry. SCG '04, New York, NY, USA, ACM (2004) 253–262
8. Blundo, C., De Cristofaro, E., Gasti, P.: Espresso: Efficient privacy-preserving evaluation of sample set similarity. In Di Pietro, R., Herranz, J., Damiani, E., State, R., eds.: Data Privacy Management and Autonomous Spontaneous Security. Volume 7731 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 89–103
9. Zhang, J., Lu, H., Lan, X., Dong, D.: Dhtnil: An approach to publish and lookup nilsimsa digests in dht. In: High Performance Computing and Communications, 2008. HPCC '08. 10th IEEE International Conference on. (Sept 2008) 213–218
10. Jianzhong, Z., Boyang, Y., Hongbo, L., Xiaofeng, L.: Peernil: An approach to publish and lookup nilsimsa digest in chord. In: Communications and Networking in China, 2008. ChinaCom 2008. Third International Conference on. (Aug 2008) 202–207
11. Azab, A., Layton, R., Alazab, M., Oliver, J.: Mining malware to detect variants. In: Cybercrime and Trustworthy Computing Conference (CTC), 2014 Fifth. (Nov 2014) 44–53