Towards Migration of User Profiles in the SONIC Online Social Network Federation

Sebastian Göndör, Felix Beierle, Evren Küçükbayraktar, Hussam Hebbo, Senan Sharhan and Axel Küpper
Service-centric Networking
Telekom Innovation Laboratories, TU Berlin, Germany
Email: sebastian.goendoer@tu-berlin.de, beierle@tu-berlin.de, evren.kuecuekbayraktar@campus.tu-berlin.de,
hussam.hebbo@campus.tu-berlin.de, senan.mh.sharhan@campus.tu-berlin.de, axel.kuepper@tu-berlin.de

Abstract—**As of today, even though there is a strong trend of Online Social Networks (OSNs) becoming the main communication medium, OSN platforms are still mostly proprietary, closed solutions, which are not capable to seamlessly communicate with each other. The research project SONIC (SOcial Network InterConnect) proposes a holistic standard for inter-platform communication to eradicate these gaps between OSN platforms. Yet, even with such a communication architecture, users are still bound to the platform they originally signed up with. We envision a mechanism that allows users to migrate their social profiles between OSN platforms at any time without losing any data or connections. To facilitate a seamless migration of a user's social profile from one OSN platform to another, we propose a standardized container format for social profile migration and a protocol for migration of the profile data. This allows users to move their social profiles to a new platform server without losing any data such as images or status messages. In order to uniquely identify social profiles across multiple platforms, a globally unique identifier (*Global ID*) is assigned to each profile. This way, social profiles, as well as references to such profiles can be kept intact when the location of the social profile has changed due to migration. To inform linked social profiles about the recently conducted migration of the profile, a Global Social Lookup System (GSLS) maintains a database of *Social Records*, which link the *Global ID* to the current profile location.**

Keywords–Online Social Networks; Social Profile Migration

I. Introduction

Online Social Networks (OSN) have become an integral part of our everyday digital social lives. While functionality of early social platforms, such as *Classmates.com* or *Sixdegrees* was mostly limited to discussion boards and modeling and maintaining relationships to friends and colleagues, today's OSNs have become one of the main communication platforms, which allow users to communicate via text, audio, and video, share content, plan events, or just stay in contact with friends and relatives. While a large number of competing OSN platforms with a broad variety of features exist as of today, *Facebook*, which was founded in 2004, managed to overcome its predecessors and competitors by far in terms of number of users and popularity [1]. This forced many competitors to discontinue their services, or focus on niche markets, such as focusing on modeling links between business partners.

Today's OSN platforms are mostly organized in a centralized manner. This forces users of OSN platforms to not only entrust all personal information to the respective platform's operator and surrender copyrights of the profile's contents to the platform's operator, but also creates lock-in effects, so users are bound to the OSN platform they registered with. These lock-in effects are used to keep users from migrating to other OSN platforms at a later time. Personal data acquired from the users is then used e.g. for targeted advertisement, giving the users little or no control over how and what information is used [2][3][4]. Decentralized OSN alternatives such as the open source applications Diaspora *Diaspora* [5] or *Friendi.ca* [6] allow users to host their own data at an arbitrary server. Still, these approaches fail at allowing seamless communication with arbitrary other OSN platforms [7].

The research project SOcial Network InterConnect (SONIC) proposes a holistic standard for social inter-platform communication. Here, a common protocol is used to allow different kinds of OSN platforms to interact directly, while gaps between different platforms and servers, i.e., the fact that a social profile is hosted at another OSN platform, are kept hidden from users [8]. Following this approach, OSN platforms support a common API and protocol, which allows to exchange social information across platform borders, while addressing remotely hosted user accounts directly via a globally unique user identifier. The result is an *Online Social Network Federation* (OSNF), defined as a *heterogeneous network of loosely coupled OSN platforms using a common set of protocols and data formats in order to allow seamless communication between different platforms* [8]. If a user requests social content such as status updates, messages, or images from another user, the required data is retrieved from the social platform server of the targeted user and displayed directly in the user interface of the requesting user's social platform. As a result, users do not need to be aware of the fact that their friends might be using a different type of OSN. Besides the lack of a common communication protocol for OSN platforms, a standard for migrating social profiles between different OSN platforms has not yet been proposed. Such a standard would allow users to export their social profiles to another OSN platform if e.g., the terms and conditions of the former platform operator are changed. In this paper, we present ongoing research on a migration mechanism that allows users of any OSN platform in the SONIC OSNF to migrate their social profiles to another OSN platform of their choice. Connections between social profiles are kept intact and therefore allow a seamless handover.

In this paper, we present ongoing research on a migration mechanism that allows users of any compatible OSN platform in the SONIC OSNF to migrate their social profiles to another OSN platform of their choice. Connections between social profiles are kept intact and allow a seamless handover. In Chapter II, an overview of existing standards is provided. Chapter III describes details of the migration mechanism, while Section IV concludes the paper.

II. Related Work

Several existing OSN platforms offer functionality that allows a user to export and download parts of or even the complete social profile. The motivation behind this kind of export functionality is rarely to allow the social profile to be moved to a different OSN platform, but rather to enable users to create a local profile backup. For example, Facebook, Google+ and Twitter offer export functionality that saves accumulated data into an archive, which can be downloaded by the user. Other networks such as LinkedIn only offer mechanisms to export contacts as a .csv or .vcf file, yet an export mechanism for the whole profile is not provided. Federated and open source approaches such as Friendi.ca or Diaspora also offer basic data export mechanisms. Even though profile data can be exported, the functionality has been designed for a personal data backup for user profiles and lacks an import mechanism [9]. The DataPortability Workgroup that aimed to define data formats and best practices based on open formats to exchange user account data between different platforms [10][11]. In 2008, Facebook and Google joined the Workgroup, however, as the present lock-in situation demonstrates, to no avail.

For identification of user profiles, current OSNs mostly use a locally unique identifier in combination with the platform server's domain name. Here, URLs (e.g. http://osn.com/alice) or email-like identifiers (e.g. alice@osn.com as with XMPP, Diaspora, or Friendi.ca) are the most common formats. The resulting identifiers are globally unique, but bound to the platform's domain name. Hence, connections to other social profiles would be lost when migrating a profile to another server, as the user identifier needs to be changed to reflect the change in location. Universally Unique IDentifiers (UUIDs), are a general format concept to generate 128-bit globally unique identifiers in a distributed fashion. Here, hash functions are used to reduce the probability of a collision to a minimum, therefore allowing for UUID generation without a central entity or directory [12]. Similar approaches such as Twitter Snowflake or boundary flake have been proposed, which are based on the same principle. As these IDs do not comprise location information, additional directory services are required to resolve a UUID to a URL.

Even though several OSN platforms allow exporting parts of a user's social profile, functionality for uploading a previously downloaded social profile is missing, thereby rendering it impossible to migrate or restore social profiles. Furthermore, user identifiers are mostly bound to a certain domain, which would result in a loss of links to other social profiles when the location of a profile changes. To this point, a holistic mechanism that allows migration of complete social profiles has not yet been proposed.

III. Profile Migration

Implementing inter-platform communication through standardized protocols and data formats as proposed by SONIC allows users of OSN platforms to freely choose, which platform operator they want to entrust their data to. However, once set up at a certain platform, a social profile cannot be moved. Although some OSN platforms allow to export (parts of) a user's profile data, importing this data into another OSN platform is usually either impossible or cumbersome and has to be done manually. Moreover, the most severe drawback is that links to other users are lost, as a change of the OSN platform results in a different URL of the profile. In order to truly eradicate lock-in effects of today's OSN platforms and allow users to become fully independent of any OSN provider, users need to be able to migrate their social profiles between different OSN platforms in a standardized and automated fashion without losing connections to and from other social profiles. This way, users are enabled to reconsider their choice of an OSN platform and move their social profiles, if for example the OSN platform's terms of usage service are changed, or the platform operator goes out of business. The research project SONIC proposes a set of social container formats, which are designed to store social profile data including profile pages, exchanged messages, status updates, or images. By providing a common protocol for extraction of a social profile as well as importing the data on any compatible target OSN platform, social profiles can be migrated between platform servers automatically. To further allow links between different social profiles to be kept intact when migrating to other servers, SONIC proposes to assign a *Global ID* to each social profile. *Global IDs* are domain independent and globally unique identifiers that allow to address social profiles independently of their URL. Information about the current location of a social profile is specified in a *Social Record*, which links the *Global ID* to it's actual URL. Finally, all *Social Records* are published and stored in the Global Social Lookup System (GSLS). The GSLS is a decentralized and global directory service, that allows OSN platforms to resolve a *Global ID* to retrieve the current location.

A. Global Identifiers

Social profiles are highly complex datasets, which are interlinked with each other. Traditionally, social profiles are identified via a username, which is unique only for the hosting OSN platform. In combination with the OSN platform's domain name, a globally unique, but domain-dependent user id is created. Therefore, changing a social profile's location would break links to other users' social profiles as a result of the changed profile location. This problem extends beyond the friend rosters, as social profiles are commonly linked in content such as status updates, conversations, or images. Therefore, SONIC proposes the use of a *Global ID* as a domain independent and globally unique identifier, which is kept intact and unchanged during the migration process. As depicted in Figure 1, the *Global ID* is a 256-bit sequence created by using PBKDF2 [13] with 10,000 iterations using the user's account public key, and a salt of fixed length. The output is converted to base36 ([A-Z0-9]) to shorten it for the use on screen and in URLs. As *Global IDs* do not comprise information about the location of a social profile, SONIC proposes the GSLS as a global directory service, which allows to resolve a *Global ID* to it's matching *Social Record*. *Social Records* comprise information about a social profile's current location, as well a digital signature as proof of authenticity and integrity.

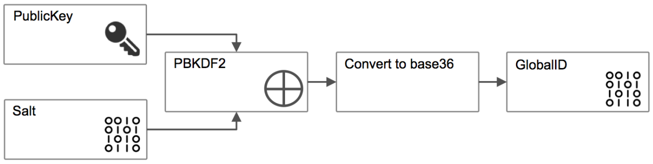

Figure 1. Deriving the GlobalID from the *Personal PublicKey* using PBKDF2

B. Social Records

Social Records are JSON encoded datasets, which contain a set of information denoting - among other data - the current social profile's location. Each Social Record dataset is uniquely identifiable by it's *Global ID*, which is used as a globally unique lookup key in the GSLS. By retrieving a *Social Record* for a known *Global ID* from the GSLS, information about the current location of the associated *Social Profile* can be retrieved. All *Social Records* are stored in a decentrally organized directory service. This directory service, the GSLS, uses a DHT to distribute the *Social Records*. When retrieving the *Social Record* for a *Global ID*, authenticity and integrity of the data can be verified by a digital signature, which is part of the *Social Record*. The public key required for verification of the signature is also included in the dataset. This way, data corruption or intentionally altered *Social Record* datasets can be detected. Each *Social Record* contains the public keys from two key pairs associated with the according social profile:

- **Personal KeyPair** The *Personal KeyPair* is used to create the *Global ID* of a *Social Account* and sign the *Social Record*. The *Personal KeyPair* cannot be revoked or exchanged, as any change of the public key would result in a new *GlobalID*.
- **Account KeyPair** The *Account KeyPair* is used and managed by the *Platform* to sign content created by the account owner as well as requests and responses. As this requires that the private part of the *Account KeyPair* is entrusted to the platform, the *Account KeyPair* can be revoked and exchanged with a new key pair. When exchanging a key pair, revocation information is published as part of the *Social Record*, which is signed using the *Personal KeyPair*.

Exchanging the cryptographic keys of the *Social Record* would allow an attacker to alter the included data and create a valid digital signature. To prevent this attack scenario, the *Global ID* is derived directly from the public key and a salt of fixed length. Therefore, exchanging the cryptographic keys of the *Social Record* would automatically alter the *Global ID*. As the *Global ID* is used as the lookup key in the directory service, an exchange of the *Personal KeyPair* is rendered impossible. In order to allow revocation of the *Account KeyPair*, the *Personal Keypair* is used to both create the *Global ID* and sign the *Social Record*. This way, the *Global ID* remains unchanged when the *Account KeyPair* needs to be revoked. Revocation information is also stored in the *Social Record* and signed using the *Personal KeyPair*. The *Social Record* comprises the following information:

- **Global ID** The *Global ID* is the identifier of the *Social Record* as well as the social profile. It is a 256-bit sequence created from the public key of the *Personal KeyPair* and a salt.
- **Salt** A sequence of 16 random characters.
- **Account PublicKey** Digital signatures for content, requests, and responses are created using the *Account KeyPair*. To allow other users to verify these signatures, the *Social Record* contains the public key of the *Account KeyPair*.
- **Personal PublicKey** The *Personal PublicKey* is used to create the *Global ID* of a social profile and sign the contents of the *Social Record*. Additionally, the *Personal KeyPair* is used for key revocation. As the signature of the *Social Record* is created using the private key of the *Personal KeyPair*, the *Personal PublicKey* can be used to verify the integrity and authenticity of the *Social Record*.
- **Profile Location** Specifies the URL of the SONIC API endpoint at which the profile is reachable. Using the SONIC protocol, the social profile and associated resources such as images and status updates can be requested via the *Profile Location*.
- **Timestamp** Denotes the date and time of the last change to the *Social Record*. Specified in XSD-DateTime format.
- **Display Name** Human-readable name of the owner of the social profile, which is used for on-screen display.
- **Key Revocation List** List of (potentially) compromised *Account PublicKeys*. Similar to the X.509 CRL standard [14], the list describes the revoked public key, date of revocation, and a code denoting the reason of revocation.
- **Signature** A digital signature created by using the *Personal KeyPair*. The signature covers the entire *Social Record*, rendering forging of the data impossible.
- **Active Flag** Specifies, whether the *Social Record* is active or not. A complete deletion of a *Social Records* is not possible to prevent reissuing of *Global IDs*. This way, creation of *Social Records* for already existing, yet inactive *Global IDs* is prevented.

C. The Global Social Lookup System

In order to publish *Social Records* for all social profiles, SONIC proposes the GSLS as a directory service. As SONIC proposes an open and decentralized architecture, the directory service is organized as a distributed database inspired by a DHT-based DNS alternative by Ramasubramanian and Sirer [15]. This approach provides a similar performance as the traditional hierarchical DNS, but is far more resilient against attacks [16]. The implementation of the GSLS is based on Java, where a Jetty server provides a RESTful interface for requests. Internally, TomP2P [17] is used to build and maintain the DHT, which is responsible for storage and replication of the *Social Records*. The external API features support for retrieving, creating, updating, and deleting *Social Records*:

- **READ** Retrieves a *Social Record* for a *Global ID* specified in the request.
- **CREATE** Allocates a previously not occupied *GlobalID* for a new *Social Record*. A correctly formatted and signed updated version of the *Social Record* for this *Global ID* has to be provided.
- **UPDATE** Updates an existing *Social Record* in the GSLS. A correctly formatted and signed updated version of the *Social Record* for this *Global ID* has to be provided.
- **DELETE** Disables a *Social Record* for a given *GID*. The *Social Record* is just deactivated in order to prevent the *Global ID* from being claimed by other users.

This interface allows not only to retrieve information about a social profile's current location, but also supports migration of social profiles. In case a profile is migrated to a new OSN platform, the *Social Record* can be updated accordingly. Requests to this profile are then automatically redirected to its new location.

D. Generic Data Formats

In order to support cross-platform exchange of social profiles and associated information, SONIC proposes generic data formats for social profiles in JSON, which have been extended and adopted for the purpose of migration. In SONIC, a standardized set of core features has been identified that covers functionality provided by the broad majority of all OSN platforms. This feature set covers all basic functionality of OSN platforms being profile pages, friend rosters, status messages, liking content, commenting on content, tagging people, and multi-user conversations, but can be extended easily to support additional functionality [8]. These data formats were designed based on previous analysis, comparison, and mapping of different OSNs' features in order to ensure the greatest possible coverage of standard OSN features.To facilitate profile migration, SONIC proposes a suitable migration data structure that holds the data that is common to most existing OSN platforms, which is based on the regular data formats of SONIC but has been extended and adopted for the purpose of migration. Here, every JSON-encoded message type corresponds to a table of this proposed data structure to ease the extraction of data from an OSN platform. To validate incoming and outgoing data, every JSON-encoded message type has a corresponding JSON schema type [18] to ensure conformity with the format as a security measure to avoid accepting malformed data.

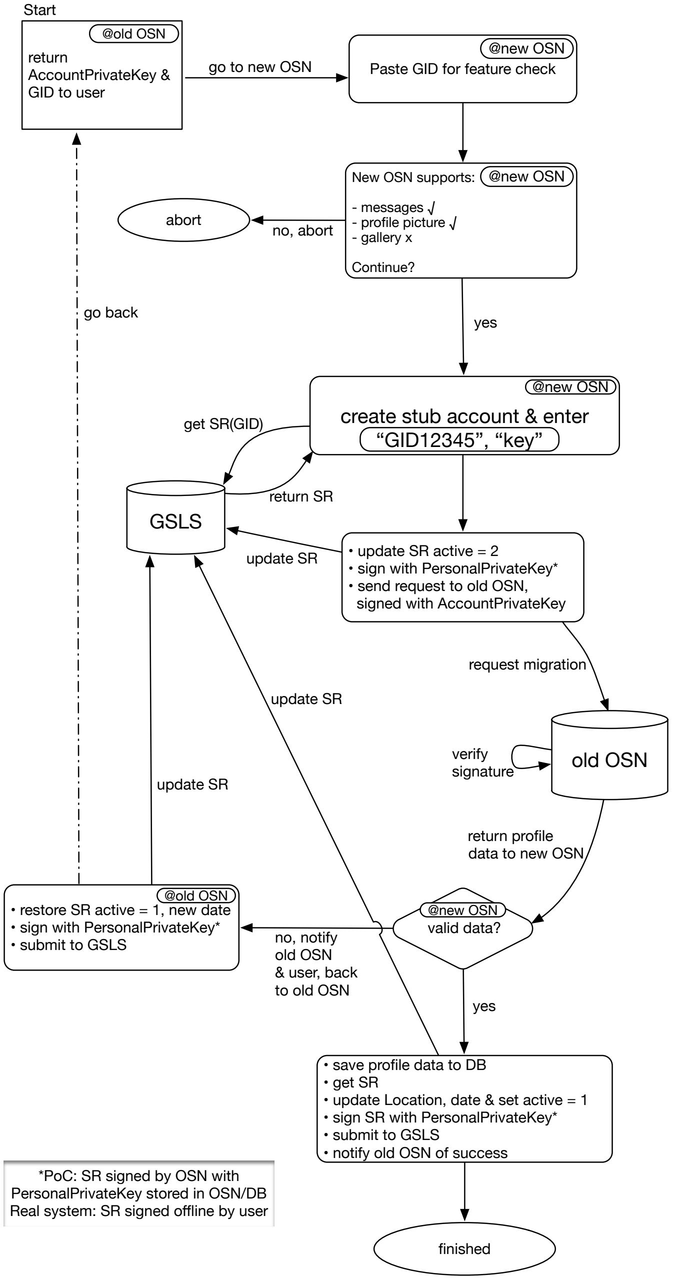
Start
@old OSN
return AccountPrivateKey & GID to user
go to new OSN
@new OSN
Paste GID for feature check
New OSN supports:
@new OSN
- messages √
- profile picture √
- gallery x
Continue?
no, abort
abort
yes
go back
@new OSN
create stub account & enter
"GID12345", "key"
get SR(GID)
return SR
GSLS
update SR
• update SR active = 2
• sign with PersonalPrivateKey*
• send request to old OSN, signed with AccountPrivateKey
request migration
update SR
verify signature
old OSN
update SR
return profile data to new OSN
@old OSN
• restore SR active = 1, new date
• sign with PersonalPrivateKey*
• submit to GSLS
no, notify old OSN & user, back to old OSN
@new OSN
valid data?
yes
• save profile data to DB
• get SR
• update Location, date & set active = 1
• sign SR with PersonalPrivateKey*
• submit to GSLS
• notify old OSN of success
*PoC: SR signed by OSN with PersonalPrivateKey stored in OSN/DB
Real system: SR signed offline by user
finished

Figure 2. Migration flow-chart: After providing both Global ID and the key pair to the new platform, all profile data is fetched from the old OSN platform using standard data formats. Finally, the profile location is updated.

E. Migration Protocol

The proposed migration protocol, as depicted in Figure 2, describes the necessary steps to migrate a social profile to a new platform. Before migrating a social profile, the user has to select a new OSN platform as a migration endpoint to which the social profile will be migrated to. At the new OSN Platform, he provides the information necessary for authentication at the old OSN platform, at which the social profile is stored. This information comprises the *Global ID* and the *Account PrivateKey*. Using the provided *Global ID*, the new OSN can now retrieve the *Social Record* of the user. Now the old OSN platform is queried for a list of supported features using `GET /features`, indicating which profile data is incompatible with the new OSN platform and therefore cannot be used or displayed after a successful migration. Only if the user still consents and confirms the migration to the new OSN platform proceeds. Otherwise, the operation is aborted. If the user agrees to proceed, the provided information is used to create a stub profile at the new OSN platform. All social profile data will be migrated to this stub profile. The user now updates the *Social Record* by setting the active flag to `active = 2` to indicate that the profile is currently being migrated and requests from other OSN platforms cannot be handled at the moment. The new OSN platform now requests the migration data from the old OSN platform using a standard remote procedure call `HTTP GET /:globalID/migration`. The request is signed with the *Account PrivateKey*, so the request's authenticity can be verified by the old OSN platform. The old OSN responds with the JSON-encoded profile data, which is validated against the specified JSON schemas and, on success, saved to the new OSNs database. Once all social profile data has been written to the new OSN platform's database, the migration is concluded by sending a notification to the old OSN platform using `HTTP PUT /:globalID/migration`, which then deletes all data. The *Social Record* is once more updated by setting the active flag to `active = 1` and updating the profile location to reflect the profile's new URL, thus concluding the migration process. If, at any moment, the migration is aborted, the new OSN platform will send a failure notification to the old OSN platform. In this case, data at the old platform will not be deleted and the *Social Record* in the GSLS will be updated by setting the active flag back to `active = 1`. Ultimately, the new OSN platform will delete all received profile data and provide a detailed error description in order to allow tracing the cause of the migration failure. To prevent misuse of the *Account KeyPair* by the old OSN platform, the key pair is revoked as an optional security measure. Upon key revocation, the new *Account KeyPair* is created by the new OSN platform, while the new *Account PublicKey* as well as the revocation certificate for the old key are published in the GSLS.

IV. Conclusion

In this paper, we described the current state of ongoing work on a migration protocol for social profiles in the SONIC OSNF. The migration protocol proposes a set of generic data formats based on widely accepted web standards to encapsulate all information related to a social profile. Using common APIs for export and import, social profiles can be migrated between compatible OSN platforms in an automated fashion. As the architecture of SONIC proposes the use of globally unique and domain-independent identifiers (*Global IDs*) for social profiles, connections to and from social profiles will not break when migrating. Profile locations are published in *Social Records*, which link a profile's *Global ID* to its current location. Profile lookup is facilitated through the Global Social Lookup System (GSLS), a DHT-based directory service. This way, connections between social profiles are kept intact even when the location of a profile is changed. The solution is currently being implemented and tested in a SONIC testbed running a specially enhanced version of *Friendi.ca* [6], which supports the SONIC protocol.

Acknowledgment

The work described in this paper is based on results of ongoing research as part of the research project SONIC (grant number `01IS12056`). The project's web page is accessible at http://sonic-project.net. *SONIC* is funded as part of the *SoftwareCampus* initiative by the *German Federal Ministry of Education and Research (BMBF)* in cooperation with *EIT ICT Labs Germany GmbH*.

References

[1] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow, "The Anatomy of the Facebook Social Graph," arXiv preprint arXiv:1111.4503, 2011.

[2] A. Datta, S. Buchegger, L.-H. Vu, T. Strufe, and K. Rzadca, "Decentralized Online Social Networks," in Handbook of Social Network Technologies and Applications. Springer, 2010, pp. 349–378.

[3] B. Fitzpatrick and D. Recordon, "Thoughts on the Social Graph," 2007, http://bradfitz.com/social-graph-problem/ [accessed: 2015-08].

[4] W3B, "Trends im Nutzerverhalten," 2013, http://www.fittkaumaass.de/reports-und-studien/trends/nutzerverhalten-trends [accessed: 2015-08].

[5] Diaspora, "Diaspora Website," 2015, http://joindiaspora.com [accessed: 2015-08].

[6] Friendi.ca, "Friendi.ca Website," 2015, http://friendica.com/ [accessed: 2015-08].

[7] A. Bleicher, "The Anti-Facebook," IEEE Spectrum, vol. 48, no. 6, 2011, pp. 54–82.

[8] S. Göndör and H. Hebbo, "SONIC: Towards Seamless Interaction in Heterogeneous Distributed OSN Ecosystems," in Wireless and Mobile Computing, Networking and Communications (WiMob), 2014 IEEE 10th International Conference on. IEEE, 2014, pp. 407–412.

[9] Diaspora, "Diaspora FAQ," 2015, https://wiki.diasporafoundation.org/FAQ_for_users [accessed: 2015-08].

[10] K. Heyman, "The move to make social data portable," Computer, vol. 41, no. 4, April 2008, pp. 13–15.

[11] DataPortability Project, "DataPortability Project Website," 2012, http://www.dataportability.org [accessed: 2015-08].

[12] P. Leach, M. Mealling, and R. Salz, "RFC4122: A Universally Unique IDentifier (UUID) URN Namespace," 2005, http://www.ietf.org/rfc/rfc4122.txt [accessed: 2015-08].

[13] B. Kaliski, "RFC 2898: PKCS# 5: Password-based Cryptography Specification Version 2.0," 2000.

[14] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "RFC5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," 2008, http://tools.ietf.org/html/rfc5280 [accessed: 2015-08].

[15] V. Ramasubramanian and E. G. Sirer, "The Design and Implementation of a Next Generation Name Service for the Internet," ACM SIGCOMM Computer Communication Review, vol. 34, no. 4, 2004, pp. 331–342.

[16] D. Massey, "A Comparative Study of the DNS Design with DHT-Based Alternatives," in In the Proceedings of IEEE INFOCOM'06, vol. 6, 2006, pp. 1–13.

[17] T. Bocek, "TomP2P, a P2P-based high performance key-value pair storage library," 2012, http://tomp2p.net [accessed: 2015-08].

[18] F. Galiegue and K. Zyp, "JSON Schema: Core Definitions and Terminology," Internet Engineering Task Force (IETF), 2013.