Peer-to-peer Content Sharing Based on Social Identities and Relationships

Paolo Bellavista, Carlo Giannelli, Luca Iannario
CIRI-ICT & DISI – University of Bologna – Italy
{paolo.bellavista, carlo.giannelli, luca.iannario}@unibo.it

Laurent-Walter Goix, Claudio Venezia
Telecom Italia – Italy
{laurantwalter.goix, claudio.venezia}@telecomitalia.it

Abstract — Mass-market users diffusely exploit remote storage solutions and centralized Online Social Network (OSN) services to share user-generated content. While these solutions are widely adopted, they open several issues related to users’ content/metadata migration on server-side components managed by third-party providers, e.g., ranging from upload traffic to privacy concerns and data ownership. We propose an original middleware to support social-driven content sharing while fully preserving content ownership. On the one hand, our middleware exploits social identities/relationships available in widespread OSN applications to seamlessly interconnect remote devices in a peer-to-peer way via spontaneous networking overlays. On the other hand, it adopts a novel social relationships model to automatically tailor inter-user content visibility. The presented prototype demonstrates the proposal feasibility in terms of both performance results and easy integration with off-the-shelf devices using unmodified HTTP/UPnP protocols.

Index Terms — Social-driven Content Sharing, Online Social Networking, Data Ownership, Middleware, Spontaneous Networking.

I. INTRODUCTION

Nowadays the digital experience is characterized by an ever increasing number of personal devices, heterogeneous in terms of hardware capabilities (e.g., CPU, RAM, screen size), software API (e.g., operating system and platform), mobility (fixed/mobile), Internet connectivity (always on/intermittent), and typical use location (at home, at work, while moving). Users exploit their devices to store and generate content, typically with different objectives in different locations. For instance, Network Attached Storage (NAS) devices in users’ home networks usually store personal pictures and data, while office desktops maintain work-related documents. In addition, mobile smartphones and tablets can both provide access to content, e.g., browsing the Web, and generate content on the way, e.g., taking pictures and videos.

Anyway, the content generated by a given device is often moved to other devices, owned either by the same user or by others, with explicit actions to be performed by users. On the one hand, the goal is to access generated data despite the currently exploited device. For instance, users aim at seamlessly rendering on their smart TVs the pictures taken by their smartphones and stored on their NAS without the burden of manually moving files among devices. On the other hand, users desire to share pictures, videos, and documents with their friends, possibly also by giving them the possibility to comment upon the shared content.

Mass-market users are already used to share content by exploiting two very popular usage patterns based on infrastructure-side components, i.e., cloud storage and Online Social Network (OSN) services. In the former case, services such as Dropbox and Ubuntu One, allow to replicate part of users’ file system on infrastructure components and to distribute it to other devices. In the latter case, centralized OSNs, such as Facebook and Twitter, support content sharing and commenting among users linked by social relationships, e.g., friendship and follower/followed. However, even if widely adopted, these solutions present some notable shortcomings:

1) data ownership – privacy concerns are widely recognized by both mass-market users and governments [1, 2]. In fact, users currently lose at least part of their control on contents in the above scenarios. Eventually, data are migrated to locations/countries with laws not ensuring users’ privacy, while profiling algorithms are commonly used to dig users’ content and prompt personalized advertisements. Moreover, users cannot be sure that their data are actually removed once they delete them or close their accounts;

2) content availability – content sharing/access strictly depend on the running status of server-side components. Even if user’s devices are active and on-line, shared content could become unavailable due to infrastructure outages, e.g., in case of natural disasters, or to account block/removal, e.g., in case users break service provider’s terms of use;

3) limited storage/bandwidth – free infrastructure-based services for content sharing are typically characterized by limited amount of storage size and up/download bandwidth, e.g., making unsuitable to store/share large files like high definition videos. Usually, fees are charged to get additional data storage and larger bandwidth;

4) service provider lock-in – users are forced to subscribe to specific service providers and exploit their applications/API to move data. While inter-provider migration is possible, it requires to manually move content between different infrastructures, both time and bandwidth consuming processes in case of large amount of data.
Given the above limitations, our goal is to support content sharing in a user-centric way, by avoiding to force users to move their data on third-party server-side components. At the same time, we aim at providing a highly usable and transparent user experience, to spread user-centric content sharing even among unskilled users. In particular, the proposed solution originally:

a) exploits social identities and social relationships to interconnect devices belonging to the same or different users. To interconnect devices, users simply have to login to their preferred OSN, thus hiding any complexity related to device configuration;

b) supports traditional social networking while sharing content in a peer-to-peer way. It takes advantage of traditional OSN capabilities to advertise and comment upon shared content; however, the shared content is directly sent from the device storing it to the device requesting it, without any storage support by third parties. Thus, it is also possible to take full advantage of increased storing/bandwidth available in home environments;

c) provides a personalized view of peer-to-peer shared content based on inter-user social relationships, e.g., allowing close friends’ access to personal pictures and colleagues’ access only to a small subset of work documents. To this purpose, we finely tune content access policies based on a novel taxonomy modeling social relationships in terms of awareness, symmetry, and mutual acceptance;

d) takes advantage of consolidated standards for service discovery and data transfer, e.g., by seamlessly integrating the UPnP and HTTP unmodified protocols with social-driven peer-to-peer connectivity mechanisms. In this way, our solution not only maintains the same content sharing experience that final users are already used to (facilitating its adoption), but also enables the seamless exploitation of legacy and off-the-shelf devices/protocols.

To this purpose, we support the creation of User Centered Networks (UCNs) and their automatic federation. On the one hand, UCNs are personal overlays based on spontaneous networking [3] tightly interconnecting devices owned by a unique social identity. Devices are typically located in different physical networks and are virtually interconnected to easily support full sharing of data belonging to the same user. On the other hand, UCN federations represent the dynamic and loosely interconnection of UCNs related to different social identities linked by social relationships. Devices belonging to different but federated UCNs interwork for content sharing, but with tailored resource visibility (see Section III) based on relationship characteristics (see Section II).

Figure 1-up shows how UCNs and their federation can be easily exploited to share content without third-party infrastructure-based services. Cate associates her Internet-connected devices to her Facebook profile, thus automatically generating a UCN composed of her tablet, gateway, and NAS. Now Cate’s tablet can effortlessly upload pictures directly to the NAS Web server via HTTP, even if the latter resides in her private home LAN. Furthermore, since Alice and Cate are friends on Facebook, their UCNs are federated, thus supporting to browse DLNA AV Media Server content stored in Cate’s NAS from Alice’s smart TV as if they were in the same IP subnet, by exploiting legacy mass-market solutions based on standard UPnP.

Let us stress that the target scenario is characterized by devices located in different IP subnets managed in a completely decentralized and uncoordinated way, e.g., NAS/desktops in home LANs of different users together with mobile terminals connected to the Internet via operator-managed UMTS networks. In fact, the proposed solution takes advantage of spontaneous networks [3] composed of different private subnets with heterogeneous IP address spaces. To this purpose, we rely on our Real Ad-hoc Multi-hop Peer-to-peer (RAMP) middleware, supporting the seamless dispatching of packets among collaborative nodes (http://lia.deis.unibo.it/Research/RAMP/). Due to space constrains we do not detail here our RAMP middleware (see [4, 5]); it is sufficient to report that it supports packet dispatching in a DSR-like way based on traditional IP addressing, not requiring modifications of operating system routing tables and network equipment modifications, thus making easier and rapidly leveraging its adoption on heterogeneous off-the-shelf devices. In addition, it supports peer-to-peer service provisioning/discovery, e.g., to dynamically retrieve file sharing services in federated UCNs and to exploit HTTP/UPnP in multi-hop spontaneous networks composed of heterogeneous IP subnets.

II. SOCIAL RELATIONSHIP MODEL

We propose an original and simple model to stress some basic characteristics of social relationships; the main goal is to automate communication/access decisions on shared content (see the following) based on social relationships and independently from any specific OSN. In our proposal, social relationships are modeled as a directional graph: vertices are entities (usually people), edges are relationships among entities
(typically relationships between people or actions related to multiple people). Figure 1-down presents four entities (“Alice”, “Bob” “Cate”, and “Daisy”) and five different relationships (“friend of”, “writes post”, “follows”, “knows” and “eng @ UniBO group”) showing that a) there is mutual friendship between Cate and Alice, b) Cate wrote a post on Bob’s profile, c) Daisy is a follower of Cate, d) Alice knows Daisy, and e) Alice and Bob attended the same college.

We claim that, independently of any specific OSN application, social relationships among entities present three main characteristics:

- **Awareness**, specifying whether both entities involved in one relationship are aware of it. In the positive case, both entities know that a relationship exists (Cate is aware that Daisy is a Twitter follower of her); otherwise only the entity creating the relationship is aware of its existence;
- **Symmetry**, declaring if the relationship is either valid on both directions or not. In the former case the edge is undirected (Anna and Bob attended the same college); otherwise the relationship is valid only in one direction and the edge is directed;
- **Mutual acceptance**, stating whether the relationship should be explicitly agreed by both entities. In the positive case the relationship must be accepted by both entities (the Facebook-based “friend of” relationship between Anna and Cate exists only if both agree).

To better explain awareness, symmetry, and mutual acceptance characteristics, consider the following examples of mechanisms involved in social relationship specification (Table 1):

a) **vCard** (http://tools.ietf.org/html/rfc6350), to manage a client-side address book. Since address books are typically managed in a completely private and client-side way, vCard specifies relationships not based on awareness, symmetry, or mutual acceptance;

b) Friend of a Friend (FOAF) (http://xmlns.com/foaf/spec/), to specify users’ personal interest and relationships as RDF triples. FOAF documents can be put on the Web and their URLs published, allowing remote users to access/process them. For this reason, relationships specified in FOAF documents can be based on awareness or not (public and private FOAF document respectively) and can be symmetric or not, but in any case do not require mutual acceptance;

c) **Facebook friendship** is based on awareness, mutual acceptance and symmetry while a **Facebook group** is based on awareness and symmetry, but does not require mutual acceptance;

d) **Twitter follower/followed** relationship is not symmetric and usually not based on mutual acceptance (default Twitter configuration does not impose to accept a new follower explicitly), but is always based on awareness (followed users are always able to get the list of their followers).

<table>
<thead>
<tr>
<th>Table 1. Characteristics of some social relationships.</th>
</tr>
</thead>
<tbody>
<tr>
<td>vCard</td>
</tr>
<tr>
<td>Awareness</td>
</tr>
<tr>
<td>Symmetry</td>
</tr>
<tr>
<td>Mutual acceptance</td>
</tr>
</tbody>
</table>

III. MAPPING SOCIAL IDENTITIES AND RELATIONSHIPS TO UCNs AND UCN FEDERATIONS

Our novel solution supports the dynamic creation of UCNs and their social-driven federation. UCN federation is based on social relationships already available on widespread OSN applications. Moreover, the awareness, symmetry, and mutual acceptance characteristics are taken into consideration to finely tune inter-user shared content visibility. Thus, even unskilled mass-market users can fully comprehend and suitably adopt visibility rules intuitively, by simply mapping social relationships with content sharing rules. In particular, our proposal identifies three different mappings:

- **Identity Mapping**, creating UCNs based on the association of one or more OSN profiles and devices to a unique social identity. For instance, the “Alice” Facebook profile and the “@Alice” Twitter profile could be associated to “Alice” UCN, managed by her home gateway and composed of her office desktop and Android smartphone;

- **Relationship Mapping**, federating UCNs based on OSN relationships. For instance, “Alice” and “Cate” UCNs are federated since their identities are in a friend relationship on Facebook. For this reason Alice’s and Cate’s devices can easily share content, as long as their friendship lasts;

- **Access Mapping**, creating visibility rules among federated UCNs by associating relationships with shared resources. Different relationships grant differentiated access to the resources of other users’ UCNs.

The main characteristics of social relationships influence our UCN federation and shared content visibility rules. In particular, we discriminate among container and content visibility: the former represents a service supporting content sharing, e.g., a shared directory or a DLNA AV Media Server, the latter is the shared content itself, e.g., a text file or a picture. In the basic case of a social relationship not based on awareness, the UCN discovering the new relationship is in charge of requesting a federation to the other identity’s UCN. This kind of relationship can be regarded as very loose because relying on the knowledge statement of only one of the involved identities; thus it typically allows to access only content with well-known locations, e.g., an URL of a picture published in a Web page or a FOAF document, while dynamic container discovery is not allowed. Instead, in case of awareness, the UCN receiving the federation request can check the relationship validity, e.g., verifying a Twitter follower/followed relationship and enabling content sharing only to followers.

Relationships characterized by symmetry usually involve a peer link among identities, e.g., since they have joined the same social group with the same role (both UCNs can trigger the federation). Users linked by symmetric relationships have the same visibility of federated UCNs and provide mutual
discovery/access capabilities to containers and their content. For instance, co-workers involved in the same project can reciprocally share content joining the Facebook group “ProjectX” and tagging a local container as shared among ProjectX participants. The same does not apply in case of asymmetry; for instance, Twitter follower/followed UCNs are federated, but while followers can discover containers in followed UCNs, the vice versa is not allowed.

Finally, mutual acceptance usually refers to tighter identities links because users have to explicitly agree on relationship requests. Therefore, in this case automatic access rules are more permissive, allowing to dynamically discover a wider set of containers. For instance, users could use the default to grant photo album access to their Facebook friends.

To provide more differentiated and fine-grained visibility of shared resources, then we additionally adopt two social-aware filtering mechanisms for tuning container/content visibility:

- **Discovery Filter (DF)** – DF modifies container visibility based on the specific request social identity. In other words, it tailors the capability of browsing shared containers, e.g., personalizing the subset of shared photo albums based on inter-user relationships;

- **Content Filter (CF)** – CF controls access to each shared content. Compared with DF, CF supports finer-grained access control, e.g., by eventually dropping unauthorized requests to specific files in a photo album.

To better explain DF and CF mechanisms, consider the following examples based on our UPnP proxy solution [5] that extends the traditional UPnP protocol to multi-hop spontaneous networks:

- users exploit DF to specify which UPnP services belonging to their UCN can be dynamically discovered by other users, providing different access rules for different social relationships. For instance, DF can grant visibility to DLNA AV Media Servers located in the home subnet to Facebook friends, while Twitter followers cannot discover them;

- users exploit CF to specify that only some files hosted on a remote Web server should be accessible. For instance, CF grants access to the user’s FOAF document to every remote user, while only Twitter followers can request pictures stored in the “twitted pictures” directory.

It is worth noting that neither DLNA AV Media Servers nor Web servers require any modification/configuration to enable the above functions: it is our middleware that transparently drops discovery/content requests not passing the applied filtering checks.

**IV. SYSTEM ARCHITECTURE AND PERFORMANCE INSIGHTS**

Based on our novel social relationship model and mapping solutions, we have developed and widely tested UCN Manager, a Java-based solution supporting UCN management and federation, built on top of RAMP. UCN Manager is activated on RAMP nodes with direct Internet connectivity, e.g., Alice’s gateway and Cate’s gateway/tablet in Figure 1-up. It is in charge of monitoring associated identities, triggering UCN federations in case of new relationships, and supporting inter-device remote communication. In particular, UCN Manager consists of three main components: Social Observer for Identifi-
ing-going from/to nodes of federated UCNs whose owners are linked by a relationship for which the local user has specified a visibility rule. The third applies to packets related to social relationships for which the user has not provided any specific rule. Users exploit our Resource Locator to dynamically retrieve available containers/content and to specify access rules based on social relationships. Based on that, Access Manager configures DF and CF, by adopting a syntax based on regular expressions. Note that while expert users can directly specify rules as regular expressions, most mass-market users are expected to use the Resource Locator GUI to map relationships with permissions simply by selecting them from drop-down menus.

To illustrate the above functionality with a practical example, consider the case of the RAMP File Sharing service, allowing to dynamically discover remote File Sharing opportunities, retrieve the set of shared files, and download the interesting ones. Access Manager may be configured with:

- DF-Facebook-friendship: "AliceNAS"
- DF-Generic            : ""
- CF-Facebook-friendship: ".+jpeg|.+mp3"
- CF-Generic            : "cv.pdf @ AliceNAS"

In case of personal UCN, no filtering is performed. For Facebook friendship, a remote user can dynamically discover the File Sharing service on top of AliceNAS device and download only jpeg/mp3 files. Otherwise, discovery and file list requests are dropped, while content requests are allowed only for the “cv.pdf” file on the AliceNAS device. Since discovery is disabled, access to this file is possible only if its URL has been explicitly advertised, e.g., publishing it in a public Facebook wall. In any case, Facebook could be exploited to advertise its URL, the actual file continues to be stored in the user’s home network and directly shared in a peer-to-peer way. Similar mechanisms are supported to access Web content and UPnP services, based on our HTTP/UPnP Proxy solution [5].

We have widely tested our Java prototype in different service scenarios and in different deployment environments. Here, for the sake of briefness, we report only two relevant sets of performance indicators, i.e., the time required to perform a federation procedure and the scalability of our filtering mechanisms. Test scenarios consist of 5 nodes (Intel Core 2 Duo 2GHz, 4GB RAM) deployed in different geographical locations with ADSL connectivity (512/9160 kbps up/download bandwidth).

Federation procedures are likely to happen not very frequently, since friendship generation is a relatively rare event (friendship revocation is similarly supported); however, users expect to access files shared by a new friend briefly after the new relationship has been made. We have tested the federation procedure in case of a new Facebook friendship: once Social Merger of any of the two users has discovered the new relationship, it starts the federation procedure, which is completed when every UCN Manager of involved users has successfully received the symmetric key and the set of federated devices. Considering 10 runs, on average it takes 3.1s (0.165s standard deviation) for the federation procedure to complete: 0.5s to connect with the remote node, 1.5s to exchange credentials, and 1.1s to spread information about keys and devices. In case of newly added UCN Managers, it requires only 2.5s (0.128s standard deviation) to notify already federated nodes.

Finally, to test the UCN Manager scalability, we have verified the processing overhead imposed by Access Monitor, i.e., the time UCN Managers take to locally filter packets. The testbed is composed of 100/400 users, each one with 2 devices. Figure 2-down shows that UCN Manager can filter packets very efficiently: the average time increases almost linearly in relation to packets/s, with successful filtering up to the rate of 1500 packets/s, and without relevant dependence on the number of connected users.

![UCN Manager architecture and performance](image)

**Figure 2.** UCN Manager architecture (up) and performance (down).

**V. RELATED WORK**

A few research activities have already investigated the exploitation of social interactions among users to easily support inter-device resource sharing. [6] compares several solutions exploiting social information, e.g., social interactions inferred by previous inter-device encounters, to optimally route packets in delay-tolerant networks. Other solutions apply interest-based similarity algorithms to improve communication effi-
ciency in peer-to-peer systems, e.g., generating clusters of mobile devices [7] or tuning file sharing discovery/advertising mechanisms [8].

Moreover, recent contributions recognize that the exploitation of centralized OSNs may harm personal data privacy and ownership. For instance, [9] proposes the adoption of distributed OSNs where users' nodes act as social peers, e.g., managing social relationships and storing personal data. [10] has specifically addressed security concerns when joining peers in distributed OSNs. However, the Diaspora initiative (http://diasporaproject.org/), together with widely accepted standard such as OAuth 2.0 and Activity Streams (http://activitystrea.ms/head/activity-schema.html), demonstrates the community interest in distributed OSNs and interoperability, mass-market users are still widely bound to traditional centralized OSNs. Our solution does not modify current users’ experience, by integrating with and enhancing traditional OSNs, while increasing users’ data ownership.

Finally, centralized OSNs have been exploited to improve resource provisioning with specific objectives. For instance, [11] exploits inter-user social relationships gathered on OSNs to optimize content delivery in peer-to-peer networks, e.g., by taking into consideration social contacts/interests when allocating flows, managing network topology, and scheduling packet transmission. Instead, [12] exploits trusted social relationships to improve privacy and security in peer-to-peer networks, e.g., by prioritizing paths involving nodes whose users are in direct social relationship.

VI. CONCLUSIONS

The paper demonstrates the feasibility of social-driven peer-to-peer solutions, achieving the twofold objective of easily supporting inter-device content sharing while avoiding the migration of personal data towards third-party server-side components. Moreover, it easily maps social relationships to shared content visibility rules, allowing mass-market users to provide differentiated access to their resources easily, based on existing inter-user relationships.

The encouraging results already achieved are stimulating our further research work on a more sophisticated social relationship evaluation, to better support the personalized tailoring of shared content visibility. In particular, we are working on lightweight techniques to estimate the tightness of inter-user links based on OSN data and smartphone-based activity inference.