Email Clients as Decentralized Social Apps in Mr. Privacy

Michael H. Fischer T. J. Purtell Monica S. Lam

Computer Science Department,
Stanford University, Stanford, CA 94305
{mfischer,tpurtell,lam}@cs.stanford.edu

Abstract. This paper proposes Mr. Privacy, a social application framework built on top of email, that encourages open competition and provides privacy for users. Applications built on Mr. Privacy are "social apps" that look nothing like email. Email is used only as a transport mechanism and personal database. We choose email because it is more pervasive than any social network and it uses standardized open protocols enabling inter-operability across vendors. Consumers can pick their email providers or even host their own servers. We have developed a prototype Mr. Privacy platform for the Android, iOS, and Firefox. On top of Mr. Privacy, we created applications which share GPS locations, music playlists, and contextual discussions of websites. Preliminary results suggest that the email protocols suffice for building these kinds of social applications. This model supports data privacy and ownership, and facilitates inter-operability and competition.

Keywords: distributed social network, privacy, email

1 Introduction

Online social networking is no longer just about visiting a social network webpage. We can now socialize *in situ*, which means that our friends are with us on online while we browse the web, play music, and play a games. Facebook is a leader in *in situ* social networking; their social plugins enable websites to gain access to Facebook's over 600 million active users. More than two million websites have integrated with Facebook, and over 250 million people engage with Facebook on external websites.

In return for access to the large social graph, companies are sharing their customer relationships with Facebook and giving the social networking partner detailed knowledge of our browsing history and *everything* that we wish to share with our friends. Social networks are now brokers of highly-marketable detailed profiles of large numbers of users[1].

All the social networks available today are *social intranets*; individuals must sign up to the same proprietary network before they can interact socially. The owner of a social intranet controls the application platform; it can decide which

applications are allowed and can demand compensation for the use of its social graph. Because of network effect, there may one day be a single monopoly owning the majority of the world's social graph. Lack of open competition stifles innovation and offers consumers less choice. In a *social internet*, there is no single owner of the global social graph. With open competition, vendors may differentiate from each other by offering EULAs (end-user license agreements) that respect privacy, or offering paid services that allow users to opt out of profiling. Application developers are free to innovate without being subjected to the control of a single-party application platform owner.

1.1 Open Social Sharing

A social network offers many important functions such as discovering friends, "stalking" people of interest, status broadcast, photo sharing, and selective interactions with friends like playing games and sharing favorite web pages. This paper focuses on the latter and asks if social applications can be shared by friends using different service providers, no different from how inter-operability is provided by telecom and email providers today. Open standards encourage competition, promote innovation, offer consumers choice, and generally lead to wider adoption. Imagine an open standard for playing music: friends can pick their service providers and music providers of their choice without having to sign up with a Big Brother social network which intercepts all interactions.

It is not easy to create an open social application platform that can challenge the status quo. To attract developers, such a platform needs a large user base and a programming abstraction that is as capable and easy to use as the comparable centralized API. It is not possible to start such a platform from scratch–social networking is sticky, individuals cannot change their network on their own and still interact with their friends. Furthermore, while it is desirable to let people own their data if desired, the solution must also accommodate the general public who would give up data ownership for free services.

1.2 Email as a Distributed Back End

This paper presents what we believe is a plausible solution to this thorny problem. We propose to build this platform on top of email. Email is a mature, scalable, and open infrastructure used by over 1 billion users. We can communicate with anybody as long as we know his or her email address. We need not sign up to join the same social network. All the shared information is stored as email messages. While most people get their email accounts from a few large companies, individuals and corporations do have the freedom to use paid, advertisement-free email services or to run their own servers.

We have created a prototype of such a system called Mr. Privacy. Mr. Privacy includes a collection of APIs to support social networking functions. Applications can get access to a user's social contacts and interact with them using simple data access operations for application-defined data types. Mr. Privacy hides the low-level details by translating these operations into email protocols, SMTP (Simple

Mail Transfer Protocol) [15] and IMAP (Internet Message Access Protocol) [7]. With Mr. Privacy,

1. users do not know that Mr. Privacy applications are email clients because they sport a user interface similar to any other social applications, and
2. developers need not know they are writing email clients either. They simply create their application-specific data structures, store them, and retrieve them from a database. The database turns out to be distributed, implemented on top of email, and the developers need not worry about hosting or scaling it!

Mr. Privacy overcomes the difficulty of creating a new infrastructure by bootstrapping with email. Applications can be built using the email system as it is today, allowing users to interact with their email contacts through social user interfaces. Even though the email protocol is not optimal, this allows for experimentation with open and distributed social networking. Demonstrated success will hopefully entice email providers to collaborate in optimizing the protocol.

1.3 Contributions

This paper introduces the concept of using email to create an open and standard platform for *in situ* social networking. By turning email into a distributed database, we are leveraging a mature and open system with an even larger installed base than the largest online social network today. Not only does it support better data privacy and ownership, its openness facilitates competition that will lead to better products for consumers.

We have created a prototype of our proposed idea called Mr. Privacy that runs on three platforms: Android, iPhone, and Firefox. We have created three proof-of-concept applications. A GPS sharing application that supports location sharing without having our whereabouts tracked by a central authority. A social music application that illustrates how friends can share music playlists while getting their music from different sources. And finally, a SocialBar extension for Firefox that let friends share comments on any webpage they visit, without having to give away their browsing history and conversations to a single third-party.

While the lack of server-side support of message filtering added significant complexity to our implementation, we found that the SMTP and IMAP protocols with the appropriate extensions are adequate as data storage and transport for the class of social applications we studied.

2 Mr. Privacy Design Rationale

Mr. Privacy is designed to provide developers the primitives needed to build social applications that can inter-operate across email service providers. These primitives are built around IMAP and SMTP to allow developers to use email to store and transport information. As social computing is quickly evolving, Mr.

Privacy is designed to be open and simple, both in interface and implementation, to allow for easy integration across platforms and innovations at the application level.

2.1 Specification

An application developer interacts with Mr. Privacy using four simple API calls that are implemented across device platforms.

- SHARE: Transmits a JSON object as a Mr. Privacy message with the specified tag to a set of recipients.
- LIST: Retrieves a list of JSON objects with the specified tag newer than a certain reference object.
- GET: Retrieves a particular Mr. Privacy JSON object.
- WAIT: Waits for a JSON object with the specified tag that are newer than a certain reference object to arrive.

Mr. Privacy messages are formatted to allow applications that are built using Mr. Privacy to programmatically read the messages. To allow for server-side search, Mr. Privacy messages have the following subject header: "<message subject>[Mr Privacy][<Tag>]". The tag is the name of the application that owns the data in the message. The message body allows the data to be both humans and machine readable. The human readable component of the email is layered using html for rich email clients, such as desktop clients, and text for text-only email clients, such as mobile phones. The machine readable component of the message encodes JSON formatted data for Mr. Privacy applications to use. The layering of information is done using multiple "multipart/alternative" MIME messages. A typical email client reads through each layer of a message and displays the highest fidelity layer that it can. Additional attachments can contain other types of data such as photos. The JSON object encoding the Mr. Privacy data can reference these extra attachments to support multimedia sharing applications.

The human readable version provides a few other benefits. Firstly, users can receive information promptly even if they are not running the application. Secondly, and most importantly, it helps make social applications "viral". If a user of an application shares a piece of data with another user who does not currently have the application installed, the email message acts as an "invitation" to use the application.

The Mr. Privacy platform keeps the user's inbox free from Mr. Privacy messages. When a new message arrives in the inbox, Mr. Privacy checks the header to see if it is for a Mr. Privacy application. If it is, the message is moved into a special folder. This is done automatically as long as a Mr. Privacy application is connected to the IMAP server.

2.2 Discussion

Data privacy and ownership. The primary benefit of an email-based architecture is providing privacy and giving data ownership back to the users. A social

application in this model can be just client software. It is not necessary to have a central server that mediates all communication; the contact information is stored on the clients' machine and all the data shared are stored with the users' respective mail provider. Higher-level privacy protection is generally expected of email providers than social network providers, as a wealth of private information is stored in email today. Note that users will be sharing their information with the vendors of their friends' choice as well, they still need to be vigilant about not sending confidential information to untrusted vendors. Nonetheless, having an open system allowing inter-operability is much preferred to having a monopoly that has the ability to modify the terms of use unilaterally.

Large installed user base and mature infrastructure. Applications written using Mr. Privacy can enjoy a large user base immediately and have no problems scaling by leveraging email. In addition, various services are offered around email identities. Users can have multiple email accounts to prevent others from linking activities of our different personas through avatars. OpenID lets us log in to many web services using our email identities without having to create a new account [16]. Webfinger lets us attach public metadata to our email accounts [20]. The information is stored with our email providers so no single server monitors who is retrieving the information. Webfinger will allow Mr. Privacy to transition to an optimized non-email protocol by allowing a dedicated data service to be linked to our email identities.

Standard and extensible data representation. The social data transport provided by Mr. Privacy allows for both normal social networking and long-tail applications. Consider for example a social application for patients in a medical study. Doctors could let their patients use a Mr. Privacy application on mobile phone to enable database functionality without having to set up a server thus avoiding the addition of a new dimension of HIPPA-compatible IT support requirements. Standard Facebook-like capabilities can be provided by transporting ActivityStreams [3] JSON objects via Mr. Privacy. Because the data are not owned by one provider, users are free to use any viewer application they wish. This brings up the need for access control on subsets of the structured social data that will be stored in email.

Social rules of engagement. Mr. Privacy applications, with user interfaces similar to social applications rather than traditional email clients, need not abide by the rules of engagement established for email. Mr. Privacy changes the contract for email by isolating the social data messages from the normal inbox. We also expect these applications to provide an intuitive way for users to specify a white list. Not only does this prevent spam, it also provides the socially pleasing "read it if you feel like it" paradigm.

Performance. Using Mr. Privacy, when a message is sent to multiple users, each user receives a separate copy. A centralized social network, on the other hand, needs to keep only one copy of the item. Luckily because of the presence of spam, many email providers have optimizations to eliminate duplicate email messages.

Lack of server side computation. IMAP supports only data queries and provides no other functionality typically expected of a social networking server. This makes it difficult for Mr. Privacy to handle public interactions or support friend-of-a-friend interactions. We have chosen email primarily for bootstrapping, extensions to email are expected in the future to support more social networking features. In the meantime, we can leverage existing services to overcome Mr. Privacy's deficiencies. For example, we can selectively make certain parts of our interactions public, such as sharing a photo with Flickr or status update on Twitter.

Invalidating information. Once an item is sent from one user to another, it cannot be invalidated. Invalidation is available in a centralized social network because it has control over all the data. While Mr. Privacy could send out a recall message to be interpreted by applications, users could still read their own email and gain access to the invalidating information.

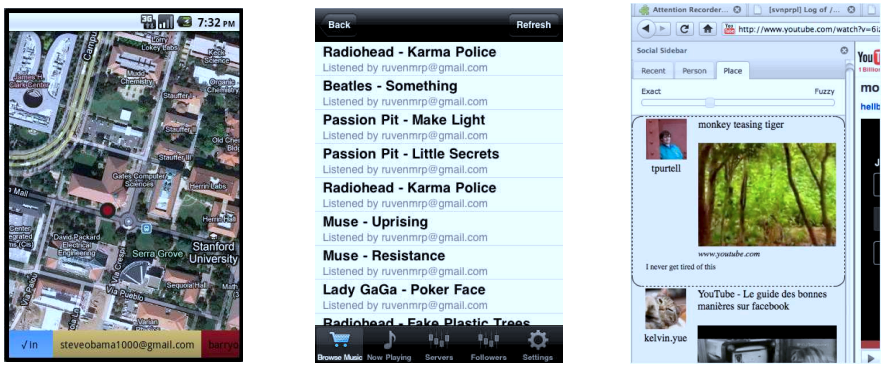
3 Prototype

We explored the implementation of a Mr. Privacy framework on three different platforms: iOS, Android, Firefox. In the ideal implementation, Mr. Privacy is a core platform service. A user trusts Mr. Privacy with with full email access, and Mr. Privacy restricts the data social applications are able to use. The Android implementation of the Mr. Privacy platform was built on the JavaMail framework, which connects directly to a mail server directly via SMTP and IMAP. Unfortunately, the iOS platform disallows local services, so each application must request credentials from the user. The Firefox browser version of Mr. Privacy is a custom mail client implemented in Javascript using the native socket transport functionality exposed to plugins.

Typical social networking applications are built on top of a centralized database. Since we cannot change email servers at all, all application-specific functionality must be provided on the clients. This fundamental shift of responsibility affects both developers and users, so we built three demonstration applications to test out the acceptability of the Mr. Privacy concept from these two points of view.

3.1 GPS Sharing

Consider physical check-in services like Facebook Places, Four Squares, and Google Latitude. While it is fun to let our friends know all the new places we are visiting, making such sensitive information like locations public is potentially dangerous. The "PleaseRobMe.com" website, for example, collects information about when people are away based on public status information and can be used by burglars to pick their victims [17]. We have created an application for the Android phone called Mr. GPS that allows individuals to exchange GPS locations using Mr. Privacy. The application sports a UI similar to other check-in services, as shown in Fig. 1(a).



(a) (b) (c)

Fig. 1. (a) GPS sharing using Mr. Privacy on an Android phone, (b) a shared playlist on Jinzora Mobile on iOS, (c) SocialBar showing comments in Firefox.

3.2 Social Music with Jinzora Mobile

As a prototype, we added playlist sharing via Mr. Privacy to Jinzora Mobile, an open-source music application on the iPhone that allows users to stream music from their PCs[12]. At 30 seconds into each song, Jinzora Mobile shares a datum with the registered friends indicating that the song is being played. Users can now select a new option "Recently Played by Friends", which uses Mr. Privacy *list* and *get* commands to fetch the list of music their friends have played. Users can view these plays, as shown in Fig. 1(b), and tap on them to start listening. By using a standard playlist format, other music players can also be built upon Mr. Privacy to share playlists across all the music services.

3.3 Contextual Social Browsing

To allow users to curate content for each other with privacy, we have developed SocialBar, a FireFox sidebar extension built on top of Mr. Privacy. SocialBar allows us to discover new content, explore a friends interests, and most importantly, discuss the pages we are viewing with friends. SocialBar provides a better *in situ* browsing experience than a portal-based one because it is built into the browser, as shown in Fig. 1(c). Our friends are available on the side as we browse the web. It is particularly engaging if we can continue the conversation on the side as we view live content. Another important advantage of SocialBar is that we can socialize on any web page. For example, academicians can discuss the content of research websites, which normally do not have any integrated social features. We can even leave notes to our friends, and ourselves for that matter, on files of a web-accessible file repository!

3.4 Suitability of Email Protocols

Deploying our applications gave several insights into the viability of the Mr. Privacy concept. We summarize the lessons learned below.

Not all users have a suitable email service. Mr. Privacy requires an IMAP service provider to allow for isolation of data messages to support the "read it if you feel like it" model. POP does not have the support for folders required to hide the messages from the user. Some providers do not offer IMAP access, but there are free alternatives like Yahoo! Mail and Gmail that will suffice.

Servers tamper with messages as a normal part of existing infrastructure. We had originally assumed that Mr. Privacy messages received would be identical to the ones sent. To our surprise, this turned out not to be the case, for example, when mails were sent to mailing lists. This can be handled by not relying on a specific MIME layout of the received messages. Also, spam detection systems occasionally quarantine messages or alter them in minor ways.

Basic IMAP does not provide the full gamut of features required. Mr. Privacy must provide alternative implementations to handle the variations in IMAP support. For example, Mr. Privacy takes advantage of the IMAP extension called "IDLE" so it can be alerted when new messages arrive in a particular folder. Similarly, Mr. Privacy uses the IMAP "CREATE" command to make a new folder on the server. Some mail services, notably AOL mail, do not support this.

Mail systems may have standard protocols, but individual implementations may have different performance characteristics. Mr. Privacy relies on existing servers and therefore must work with existing implementations. It is important that we take advantage of the techniques that servers use to optimize typical email usage. For example, Mr. Privacy takes advantage of server-side search. The first technique we explored was adding Mr. Privacy tags to message headers. This worked well for small test accounts, but would take minutes on accounts with greater than 10,000 messages. Instead, we now tag Mr. Privacy messages by adding "Mr. Privacy" to the subject and use subject search in IMAP to retrieve relevant messages. Because existing mail server implementations have an index dedicated to accelerating subject searches this search filter was dramatically faster. On large inboxes search times were reduced from minutes to seconds.

We have to make compromises in our design and implementation because we are using a legacy protocol. Nonetheless, we found that we were able to provide sufficient functionality for the kinds of social applications we have built while requiring minimal Mr. Privacy related code.

4 Related Work

A growing number of efforts are underway to provide choice in social networking. Google's Open Social allows developers to create a social application once and deploy it on different social networks [9]. However, users cannot interact across networks. This model only reduces the development effort for supporting multiple social data providers. Mozilla's Contacts abstracts existing networks

at the browser level [14] based on the W3C Contacts specification [19]. OneSocialWeb uses federated XMPP and server extensions to create a federated social network [5]. Google's Wave was a collaboration tool that used XMPP and server extensions to enable users to have rich discussion threads across providers [10]. Diaspora [11] and Appleseed [2] define a P2P protocol for social networking. PeerSoN explores building social functionality on top of distributed storage, such as OpenDHT [4].

Various other projects have proposed techniques to improve the usability and effectiveness of email. Flores et al. created The Coordinator, a messaging tool that uses structured requests to capture the essence of language thus enabling social actions [8]. Cockbrun et al. explored Mona, a novel conversation based platform to enhance email for collaborative work [6]. Rodden et al. designed Mailtrays, a system that automatically organizes and filters incoming messages to match the current needs of the user [18]. Semantic Email examined how a better user interface can be presented on top of existing email infrastructure. A management agent orchestrates sending, receiving, and reprocessing messages in order to simplify tasks for the user [13].

5 Conclusions

This paper provides an alternative social application platform to the current status quo where a single company owns all the social data. By building on top of email, our proposed Mr. Privacy platform leverages the billions of email accounts that already exist, the open email protocols that enable inter-operability, and a mature and scalable infrastructure.

Application developers can choose to write their social applications as pure client software if they wish, letting all the data be stored with the users' preferred service providers and leaving scalability issues to email providers. Users can choose an email service provider or host their own email server if they are concerned about data ownership.

We have developed a prototype Mr. Privacy platform for Android, iOS, and Firefox. On top of Mr. Privacy, we created applications which share GPS locations, music playlists, and contextual discussions of websites. There is no single third-party company monitoring all our activities. Furthermore, these applications are using the social contacts in individuals email address books and there is no third-party social network owner that has complete control of the platform.

SocialBar is available at `http://mobisocial.stanford.edu/socialbar/`. Additionally, the source code including the Mr. Privacy client library is publicly released at `https://github.com/Mobisocial/socialbar/`.

Acknowledgment

The authors would like to thank Ruven Chu for his development of the Jinzora Mobile application. This research is supported in part by the NSF POMI (Pro-

grammable Open Mobile Internet) 2020 Expedition Grant 0832820, the Stanford Clean Slate Program, and the Stanford MobiSocial Computing Laboratory.

References

1. ABC News. Microsoft Deepens Facebook Ties in Web Search Battle. `http://abcnews.go.com/Business/wireStory?id=11876359`.
2. The Appleseed Project, 2010. `http://opensource.appleseedproject.org/`.
3. M. Atkins, W. Norris, C. Messina, M. Wilkinson, and R. Dolin. Activity Streams Concepts and Representations (draft), 2010. `http://activitystrea.ms/head/json-activity.html`.
4. S. Buchegger, D. Schiöberg, L.-H. Vu, and A. Datta. PeerSoN: P2P social networking: early experiences and insights. In *SNS '09: Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, pages 46–52, New York, NY, USA, 2009.
5. D. Cheng and O. Griffin. OneSocialWeb, 2010. `http://onesocialweb.org/`.
6. A. Cockburn and H. Thimbleby. Reducing user effort in collaboration support. In *Proceedings of the 1st International Conference on Intelligent User Interfaces*, IUI '93, pages 215–218, New York, NY, USA, 1993.
7. M. Crispin. Internet Message Access Protocol - Version 4rev1, 1996.
8. F. Flores, M. Graves, B. Hartfield, and T. Winograd. Computer systems and the design of organizational interaction. *ACM Trans. Inf. Syst.*, 6:153–172, April 1988.
9. Google. Opensocial, 2010. `http://code.google.com/apis/opensocial/`.
10. Google. Wave, 2010. `http://wave.google.com/`.
11. D. Grippi, M. Salzberg, R. Sofaer, and I. Zhitomirskiy. Diaspora, 2010. `http://www.joindiaspora.com/`.
12. Jinzora, 2010. `http://jinzora.com/`.
13. L. McDowell, O. Etzioni, A. Halevy, and H. Levy. Semantic email. In *WWW '04: Proceedings of the 13th International Conference on World Wide Web*, pages 244–254, New York, NY, USA, 2004.
14. Mozilla. Contacts, 2010. `https://mozillalabs.com/blog/2010/03/contacts-in-the-browser/`.
15. J. Postel. Simple Mail Transfer Protocol, 1982.
16. D. Recordon and D. Reed. OpenID 2.0: A Platform for User-Centric Identity Management. In *DIM '06: Proceedings of the Second ACM Workshop on Digital Identity Management*, pages 11–16, 2006.
17. The Register. Burglars used social network status updates to select victims, 2010. `http://www.theregister.co.uk/2010/09/13/social_network_burglary_gang/`.
18. T. Rodden and I. Sommerville. *Building conversations using mailtrays*, pages 159–172. North-Holland Publishing Co., Amsterdam, The Netherlands, 1991.
19. W3C. Contacts. `http://www.w3.org/TR/2010/WD-contacts-api-20100121/`.
20. Webfinger, 2010. `http://code.google.com/p/webfinger/`.