Clustering and Information Sharing in an Ecology of Cooperating Agents or How to Gossip without Spilling the Beans

Leonard N Foner
Agents Group, MIT Media Lab
E15-305, 20 Ames St
Cambridge, MA 02139

http://foner.www.media.mit.edu/people/foner/

Abstract

Many future applications for advanced software agents imply distributed computation involving sensitive or private data. Most efforts to date have assumed that privacy may be traded away in order to distribute the computation, or assume that the only viable choices for users are to entrust their data completely to a third party, or not at all. This research, still very much in progress, is intended to yield a ubiquitously-deployed, distributed system on the Internet in which user privacy and doing useful work do not have to be traded off against each other.

1 Introduction

Software agents are computer programs which attempt to perform some set of tasks autonomously for their users, in a trustworthy, personalized fashion. They can be either manually programmed by the user, or "watch over the user's shoulder," using techniques from machine learning, in order to discover how the user does some task and gradually take it over from him or her. Examples include automated mail filtering programs, which learn or are told whose mail is valued and whose is not [Maes 94], [Lashkari, Metral, & Maes 94]; meeting scheduling programs, which learn or are told when and with whom to schedule meetings and how flexible to be in negotiating (with other agents) for times depending on who else is in the meeting [Kozierok 93]; and so forth. Many are even designed to be primarily entertaining, perhaps with ancillary practical or informative goals [Mauldin 94] [Foner 93].

The existence of such "standalone" agents is itself somewhat worrisome from the standpoint of individual privacy, since, in order to be effective, they must necessarily contain some amount of personal information which may often be politically or sociologically sensitive (e.g., how other people are valued for purposes of mail filtering, etc). Solutions to these problems, *if understood in advance by implementors*, are relatively straightforward (e.g., file security, conventional encryption of databases, etc).

However, many foreseeable future applications for software agents involve large numbers of agents interacting with each other [Huberman 88]. Users may have a number of agents operating on their behalf, and the agent or agents of any particular user may have to communicate with other agents elsewhere on the network in order to share information (some examples of such applications will be given later). Unfortunately, such interactions breed major problems of personal privacy and control of personal information. The answer is not to give up on communication and ignore all the advantages of networking, but rather to think carefully about how that communication is handled.

To date, most research in the field of agents has addressed problems of personal privacy strictly in passing, or not at all. This is, in part, due to the academic focus of most such research and the limited deployment of such systems, both of which tend to finesse problems of privacy by assuming very small groups of mutually-trusting individuals. However, we are starting to see systems which may involve dozens of individual agents cooperating, such as [Lashkari, Metral, & Maes 94], in which individual mail filtering agents all store information in a common database to permit group or social filtering [Shardanand 95] of the information—leaving individual users wide open to invasions of their privacy. Systems such as General Magic's Telescript [Telescript 94] do somewhat better, but are mostly oriented towards a binary model of transactions (either you have the key to the encrypted communications or not), and explicitly-safe computation (the emphasis is on preventing trojan horses, rather than safeguarding privacy).

It is quite commonly the case that new information technologies are deployed without much, if any, consideration to users' privacy concerns, and hence with little or no safeguards built into the architecture of the system. We see these trends every day, whether it is in (often legally questionable) database matching between disparate organizations' data, correlation of supermarket UPC data with individual charge card numbers, or any of a number of similar cases. It is sometimes the case (generally after some truly egregious disaster of compromised privacy) that mechanisms to protect privacy are grafted onto systems already in operation. However, it is commonly accepted in the operating system and computer se-

curity disciplines that adding privacy or security mechanisms after the bulk of the system has been designed is always a mistake—such additions are difficult to make, fragile and likely to break, and usually interfere with legitimate usage of the system to such an extent that they are routinely disabled or circumvented by even nonmalicious users.

Cooperative agent architectures are likely to suffer from the same problems, leading to lack of user acceptance (for those users who understand enough of the technology to realize the glaring lack of privacy) or surprising and unpalatable violations of confidentiality (for those users who are rudely surprised later on)—unless suitable technology is widely known and already in place for others to draw upon in implementing new systems.

To counter these trends, the work described below focuses explicitly on issues of user privacy and how it affects the architecture and design of multi-agent systems, with the aim of demonstrating that such systems are both possible and desirable.

This paper describes work that is very much in progress. The work is composed of two major parts. The first aspect (not described here) deals with cooperating ecologies of agents, and how such agents can find each other on the network and form clumps to enable specialization of knowledge, so that every individual agent does not need to know everything. The second aspect, and the primary subject of this paper, deals with protecting user privacy in such a distributed system. It discusses some important facets of the problem, touches upon some (but by no means all) of the techniques that can help, and explains how these ideas may be tested in a large, diverse set of agents which communicate on a network such as the Internet.

2 Sample applications

Consider, for example, a system which attempts to be a "matchmaking" service. Each user runs some sort of agent which knows what topics are of interest to the user, and the individual agents communicate with each other over a network to try to find users with similar interests. There might be several slightly different slants on this application:

- Coalition building: in other words, finding groups of other users who share particular professional or avocational interests. In subjects for which a tradition exists of literature, conferences, and so forth, finding others who are interested in the topic may be relatively easy. However, someone who is interested in, say, collecting stamps about yak herders may assume that no one else could possibly have such an interest, and would consequently be very interested in discovering others who do.
- A "classified ad" system in which buyers and sellers, rather than using a centralized system to post ads and scan for products (as is currently done in, e.g., newspapers classifieds) instead use a system in which requests to buy and requests to sell rove around the network, building clumps of ads about similar topics in which each part of the clump is nearby in some semantic space. Such a system might not suffer from the problems of traditional systems (in which all buyers must read all ads from all sellers), such as quadratic growth of the search space as the number of users grows, or imposed, arbitrary precategorization of ads which may or may not accurately match the attributes that users care about.
- A combination of the above, more resembling traditional romantic matchmaking, in which users and their interests *are* the ads.

Naive solutions to these problems only work when none of the information involved is sensitive to any of the participants. This seriously limits such a system, since, as the potential community of users gets larger, those same users will become cagier about what information they are willing to distribute.

In order to force the situation in this research, we are designing and implementing an architecture for doing coalition-building or matchmaking which learns what users' interests are by scanning their mail. Since most users have a high expectation of privacy for their mail, such a system must adhere to stringent requirements as to how it obtains its information, how it stores what it learns, and how it communicates with the rest of the world.

To take an extreme example, simply broadcasting users' mail to any other agents that might want to listen would be unacceptable to just about everybody. The main objection, of course, is that even if the receiving agents were *supposed* to only use the information to suggest possible matches, without revealing its details to other users, we cannot enforce such a restriction. In any network of reasonable size, we may expect both malicious agents and insecure intermediate communication (both courtesy of malicious users). Clearly, any solution must control what information can *leave* the user's agent, rather than assuming that such filtering will happen later. (We shall investigate below what sort of information might therefore be leaving an agent.)

Two immediately apparent strategies for a working design are to employ cryptography (to protect communications en route) and anonymity (to enable a user's agent to be freer in what information it might volunteer, since such information could not be traced back to the agent's user). Neither of these approaches completely solves the problem, but both deserve a slightly closer look.

Routine use of strong cryptographic protocols provides several benefits. Chief among them are *confidentiality* (protection from eavesdropping, both of messages en route on the network, and of the agent's personal database on disk), *non-repudiability* (assurance that what is said cannot be un-said), and *authenticity* (assurance that the user is who he claims to be), in varying degrees depending on the type of encryption. However, most such protocols are more suitable for "all or none" privacy, e.g., given that some agent is deemed worthy to receive some information, it gets all of it. In the case of a matchmaking agent, the whole point is that we do not know a priori who is allowed to receive certain information about the user. While cryptography is undoubtedly part of the solution, and is necessary to protect against evesdropping and alteration of data between agents, it alone cannot solve some of the hardest problems, in which a more "continuous" range of information sharing might be appropriate.

Anonymity is not a complete solution, either, for two reasons. First, it is often the case in matchmaking that the participants will, at least at some point, wish to know each other's actual identities. (We shall ignore for the moment the potential for an unequal trade of identities, e.g., if the user claims to be someone he is not—there are ways around this problem, such as the "web of trust" and "trusted introducers" employed by PGP [Zimmermann 94].) Second, we cannot assume that all attacks on an agent will follow in-band protocol. For example, if users traditionally run agents on personal workstations, then no claim of anonymity will work against a remote agent which simply checks the IP address of the incoming message—fingering at that address will yield the name of the user. While there are potential solutions to this as well (for example, employing blinded [cryptographically sealed] data that is randomly routed through a number of agents before arriving at its actual destination), it may be easier in practice to dispense with claims of anonymity and instead control what information is traded more precisely. (This has important secondary benefits as well; for example, even if no deception exists, people may still have very different preferences about what information should be "leaked" by their agents to friends versus supervisors.)

More workable solutions to the problems above assume an increasingly interactive style to the way agents relate. Agents "flirt" with each other, revealing small amounts of information at any one time, in order to try to ascertain whether their controlling users would make a good match (for whatever kind of match it is we are making). Such flirtation can take several forms:

- Interactive, give-and-take negotiations, in which each agent advances some tidbit, and waits for the other one to say something useful back. If at some point one agent decides that a match is not forthcoming, it may unilaterally leave the conversation and search for a more suitable match.
- Zero-knowledge proofs [Schnieir 94], in which two (or, if necessary, more) agents can prove to each other that they know some particular piece of information (hence meaning that their users share some characteristic), without ever actually stating what that characteristic might be. Such proofs are quite interesting—here we trade an agent's lack of human judgment for a method of communication that is infeasible for people but trivial for agents.

Any reasonable solution in applications like this must also not only be technically robust, but sociologically and psychologically acceptable as well. Even a technically reasonable solution to many of the privacy problems in such systems is unlikely to be fully accepted by the more cautious members of the user community unless they can be assured that hidden "back doors" or channels for large-scale information leakage are not present in the agents they use. The cryptographic community on the Internet has worked out several approaches to problems such as these, involving infeasible-to-forge checksums of distributed applications (to guarantee that the application the user is running is the application that claims to be distributed), and digital signatures on the checksums from trusted third parties (who have checked the source code of the application themselves and signed off on it). Barring extreme subtlety or a large-scale conspiracy, such measures can make even quite sensitive applications (including cryptographic packages themselves, such as PGP [Schneier 94] [Garfinkle 94]) safe to distribute and safe to use.

3 Where we are going

To examine these questions, we are building a toolkit allowing rapid prototyping of potential applications and agent organizations. The basic structure of the toolkit consists of a "core set" of modules which implement a basic set of capabilities that almost any networked, cooperative agent would require; the rest of the toolkit consists of the actual specialized agents that would make up some particular application. (Such modules include network-layer communications; cryptographic technology required for authentication, confidentiality, and key distribution (e.g., similiar to and based on the capabilities provided by PGP [Zimmerman 94]); scanners that can extract and parse information from, e.g., mail and other user files; correlators that look for similarities between users; one or more user interfaces; and so on.)

Particular modules in the core set are designed to be interchangeably replaced with similar modules; hence, while any given agent could depend on there being *some* implementation of each module in the core, the particular implementation of any given core module is not fixed. This accomplishes several goals. One obvious goal is to encourage others to add functionality and experiment with different applications. A more subtle goal is to permit easy swapout of export-controlled cryptographic subroutines, to enable export of the rest of the software overseas, where it may be reintegrated with the local cryptographic technology.

The entire package is designed to be freely redistributed on the Internet, with the hopes of being used in a large user community.

4 Conclusion

It is not necessary to just stand by and bemoan the increasing erosion of personal privacy as increasingly sophisticated software manipulates increasingly detailed personal information. One can use the very existence of such software and data, aided though not completely solved by the use of cryptographic ideas, to instead enhance social activities such as coalition building. The sometimes conflicting goals of utility and privacy can be reconciled if the system is designed from the beginning to protect individual privacy while enabling distributed, shared computation. These frankly political ideas, both of protecting user privacy and demonstrating that routine use of cryptography is not just for drug dealers and "people with something secret to hide", motivate the development and deployment of a toolkit for exploring these issues in the very large user community of the Internet.

5 References

[Foner 93]
"What's an Agent, Anyway? A Sociological Case Study," Leonard Foner, *Agents Memo 93-01*, MIT Media Lab, Cambridge, MA, 1993.
(http://foner.www.media.mit.edu/people/foner/Julia/)

[Garfinkle 94]
PGP: Pretty Good Privacy, Simson Garfinkel, O'Reilly & Associates, Cambridge, MA, 1994.

[Huberman 88]
The Ecology of Computation, B. A. Huberman, editor, Elsevier Science Publishers B.V., 1988.

[Kozierok 93]
A Learning Interface Agent for Meeting Scheduling," Robin Kozierok and Pattie Maes, *Proceedings of the 1993 International Workshop on Intelligent user Interfaces*, ACM Press, New York, 1993.

[Lashkari et al 94]
"Collaborative Interface Agents," Yezdi Lashkari, Max Metral, and Pattie Maes, *Proceedings of the Twelfth National Conference on Artificial Intelligence*, MIT Press, Cambridge, MA, 1994.

[Maes 94]
"Agents that Reduce Work and Information Overload," Pattie Maes, *Communications of the ACM,* July 1994, Vol 37 #7.

[Mauldin 94]
"ChatterBots, TinyMuds, and the Turing Test: Entering the Loebner Prize Competition," Michael Mauldin, *Proceedings of the Twelfth National Conference on Artificial Intelligence*, MIT Press, Cambridge, MA, 1994.
ßπ(http://fuzine.mc.cs.cmu.edu/mlm/aaai94.html)

[Shardanand 95]
"Social Information Filtering: Algorithms for Automating 'Word of Mouth,'" Upendra Shardanand and Pattie Maes, submitted to CHI-95, Denver, CO, May 1995.

[Schneier 94]
Applied Cryptography, Bruce Schneier, John Wiley & Sons, 1994.

[Telescript 94]
Telescript Technology: The Foundation for the Electronic Marketplace, General Magic, Inc, 1994.

[Zimmerman 94]
PGP: Pretty Good Privacy User's Manual, 1994.