Do You Trust Your Recommendations? An Exploration of Security and Privacy Issues in Recommender Systems

Shyong K. "Tony" Lam, Dan Frankowski, and John Riedl

GroupLens Research
Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55455
{lam, dfrankow, riedl}@cs.umn.edu

Abstract. Recommender systems are widely used to help deal with the problem of information overload. However, recommenders raise serious privacy and security issues. The personal information collected by recommenders raises the risk of unwanted *exposure* of that information. Also, malicious users can *bias* or *sabotage* the recommendations that are provided to other users. This paper raises important research questions in three topics relating to exposure and bias in recommender systems: the value and risks of the preference information shared with a recommender, the effectiveness of shilling attacks designed to bias a recommender, and the issues involved in distributed or peer-to-peer recommenders. The goal of the paper is to bring these questions to the attention of the information and communication security community, to invite their expertise in addressing them.

1 Introduction

People are often overwhelmed with the number of options available to them. To combat this information overload, many have turned to *recommender systems*: tools that use a user's opinions about items in some information domain in order to recommend other items to that user. For example, Amazon.com uses a recommender system to make personalized recommendations suggesting products that a user might like based on the products she has purchased, expressed an opinion about, or viewed.

There are a wide variety of recommender systems in use today. Some, like Amazon.com, are automated and personalized to each user, while others, such as Epinions.com's review system, are non-personalized and "manually operated" in the sense that users need to read and evaluate the reviews published on the site to reach a conclusion about an item. In this paper, we focus on personalized recommender systems that use *automated collaborative filtering* algorithms [1, 2, 3], which generate recommendations on the basis that people who have expressed similar opinions in the past are likely to share opinions in the future.

G. Müller (Ed.): ETRICS 2006, LNCS 3995, pp. 14–29, 2006.

Such recommenders require personal information from a user, and in return give personalized predicted preferences, which we also call *recommendations*.

Recommender systems require two types of trust from their users. First, since the recommender must receive substantial information about the users in order to understand them well enough to make effective recommendations, they must trust that the system will protect their information appropriately. Second, automated recommender systems are often fairly opaque to their users. Although the algorithms used are easy to understand in principle, a user is usually not presented with sufficient information to know exactly how or why an item is being recommended to her. Thus, in order for a recommendation to be accepted, the user must trust that the recommendations are accurate.

Violations of user trust in a recommender come in three flavors:

Exposure. Undesired access to personal user information.

Bias. Manipulation of users' recommendations to inappropriately change the items that are recommended.

Sabotage. Intentionally reducing the recommendation accuracy of a recommender.

Exposure. There are many examples of exposure of private user data. In 2004, hackers accessed a University of California, Berkeley system containing the names and social security numbers of about 1.4 million Californians[1]. Identifying information is expected to be kept private, but so is preference information: during Robert Bork's confirmation hearings for the U.S. Supreme Court in 1987, his movie rental history was leaked to the press. In response, lawmakers passed the Video Privacy Protection Act of 1988 making it illegal to disclose personally identifiable rental information without consent. We do not yet know of recommender information being leaked or stolen – but many companies who own recommenders are not required to publicly report identity theft. A Harris poll in 2003 finds 90% of people are concerned about protecting themselves from misuse of their personal information[2]. Ackerman et al. found 83% of people more than marginally concerned about privacy [4].

What can be done about recommender system exposure? Can security techniques from other domains be applied in unique ways to recommender systems to make privacy violations difficult or impossible?

Bias. Bias may be to increase ("push") or decrease ("nuke") the visibility of other items. In 2002, Amazon.com's page for a spiritual guide by well-known Christian televangelist Pat Robertson included an automatically generated recommendation for "The Ultimate Guide to Anal Sex for Men". "Amazon conducted an investigation and determined these results were not that of hundreds of customers going to the same items while they were shopping on the site."[3] Instead, it is likely that a few motivated people accomplished this by repeatedly viewing the two items in sequence.

[1] http://www.securityfocus.com/news/9758

[2] http://harrisinteractive.com/harris_poll/index.asp?PID=365

[3] http://news.com.com/2100-1023-976435.html

In 2004, Amazon.com's Canadian site suddenly accidentally revealed the identities of thousands of people who had anonymously posted book reviews. It turned out that authors were praising their own books and trashing other authors' books. The New York Times reported that "many people say Amazon's pages have turned into what one writer called 'a rhetorical war,' where friends and family members are regularly corralled to write glowing reviews and each negative one is scrutinized for the digital fingerprints of known enemies."[4] To increase the credibility of some reviews, Amazon now has a "Real Name" badge applied to reviews written by customers who have verified their identity and agreed to publicly reveal that they wrote the review.

How can bias be limited in recommender systems? Can information-theoretic techniques be used to identify attempts to bias recommendations?

Sabotage. There are many examples of sabotage in web sites. The most common are denial of service attacks or defacement of the front page. Besides these and some of the bias attacks mentioned above, we know of no other direct sabotage attacks on recommender systems to date. We hypothesize that sabotage may become more prevalent in the future, as business competitors use recommenders as a key business advantage. For now, though, we recommend focusing research on the other types of attacks on recommender systems.

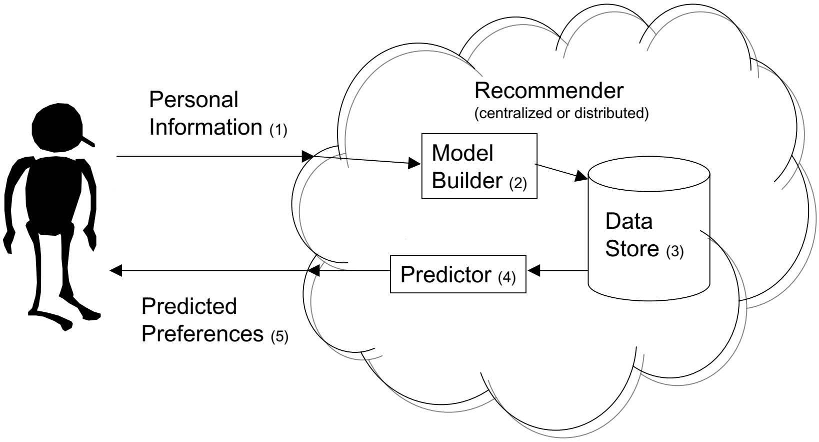

Fig. 1. Conceptual model of the interaction between a user and a recommender system

Figure 1 shows high-level data flow between a user and a recommender: the user gives personal information in return for predicted preferences. *Personal information* may be preferences or other identifying information such as name, zip code, age, gender, email, or account name. *Predicted preferences* may be a list of recommended items for the user to consider, or a predicted opinion for a given item or set of items.

[4] http://www.nytimes.com/2004/02/14/technology/14AMAZ.html

The recommender in figure 1 has internal structure. The *model builder* may select, combine, and compute a user model from personal information. The model builder may also be privacy-preserving if it discards or abstracts away from personal information. The *data store* holds the results of model building as well as any other information necessary for the application. The *predictor* uses the model to predict preferences. However, the figure is not intended to show concrete system architecture. For example, the TiVo TV show recommender puts the predictor on the user's machine, not a server [5]; peer-to-peer or distributed recommenders may distribute recommender components broadly.

Figure 1 is subject to many of the classic client-server security concerns: man-in-the-middle attacks, denial of service attacks, hacking into the recommender server(s), and so on. Such attacks are no different in the context of a recommender system than a standard client-server system, so they will not be considered in this paper. We instead focus on issues specific to recommenders.

Each of our research questions may be considered at several points along the flow of data. For example, predictions may be sabotaged or biased by users giving false information or misrepresenting their opinions (which we call *shilling*), or by the owner of a recommender altering the recommendations (e.g. to sell overstock or increase profit). Exposure may occur by looking for users' personal information directly or by trying to infer it from recommendations [6].

This paper is an extension of a workshop paper [7] that contains our thoughts about interesting research questions around privacy-preserving recommenders. In the present paper, we consider some topics that touch on these research questions: the prediction value and privacy cost of personal information (section 2), ways to bias prediction results (section 3), and distributed or peer-to-peer recommendations (section 4).

2 Value and Privacy Risks of Information

A personalized recommendation algorithm requires input from the user population in order to make recommendations. Providing more input potentially increases recommendation accuracy, but also increases the risk of unwanted exposure of personal information. Ideally, one would like to find a balance where the system is able to make good recommendations while not requiring users to give up too much information about themselves.

There is large body of prior work in this area. Many have looked at ways to preserve user privacy in recommender algorithms [8, 9] or datasets [10]. The data mining community has also become interested in privacy-preserving algorithms [11]. So far, they find that you can suppress, perturb, or generalize data with varying effects on algorithm outputs (such as recommendations) or dataset anonymity. Also, Ramakrishnan et al [6] describe a graph-theoretic model showing how information can be inferred about *straddlers* (users with eclectic tastes) by observing the recommendations made by a system.

When a user provides information to a recommender, two broad questions arise. 1) What value is gained? 2) What exposure is risked? We discuss value in 2.1 and exposure risk in section 2.2.

2.1 Value of Information

In this section, we discuss different ways of judging the value of preference information provided by the user. Value may take many forms: list-keeping, discussion, fun. For simplicity, we assume that value is increased prediction accuracy. The accuracy gained by providing information may be for the user who provided it (“value to self”), or for others using the recommender (“value to others”).

There are many issues to consider. How to judge the value of information? How much data should the recommender solicit? Which data should it solicit? Should it keep all the data? What user interface should it use? How does changing the data available affect recommendation accuracy? We devote a sub-section to each question.

Metrics. The purpose of information collected by a recommender is to differentiate a user from her peers. Some pieces of data are inherently more valuable than others in an information-theoretic sense and thus, are better at differentiating among users. For instance, knowing that a user likes the universally-popular movie “Toy Story” reveals less than knowing that she likes “Fahrenheit 9/11,” which has a higher level of diversity among users’ opinions. This is the basis of an idea proposed by Pennock and Horvitz that says if one can calculate how useful a given piece of information is (a *value-of-information* or VOI metric), then one can tune a system to optimize its data collection process by soliciting user preferences on items that have the most value [12].

Such a metric has a variety of uses including judging whether the recommender has enough bits of data for a particular user or movie, or directing users to provide high-value information for others.

Amount of Data. How much data is needed from a user to make good recommendations to that user? Providing a recommender with data may produce diminishing returns. That is, perhaps once a certain amount is known about a user, obtaining further information is only marginally useful. Perhaps there is a “sweet spot” that maximizes the recommendation accuracy per unit of information known about the user.

How do we calculate the sweet spot? It is desirable for the recommender to know this so that it can stop soliciting data from a user once it has built a sufficiently good user model. With a VOI metric, it is possible to measure how much information is needed to make good recommendations, and then to stop collecting new information from the user once that point is reached. More generally, a recommender system might use VOI to bound the amount of information collected about a user to some optimal level with respect to both privacy and recommendation quality.

One issue is that how much information is useful may change over time. As the system or the user evolves, or as new items are introduced, will more information be needed to maintain high-quality recommendations?

Finally, suppose it would be useful for many other users if a particular user were to give more information than necessary for her own prediction accuracy? How do we appropriately balance these competing goals? Many users are probably willing to give value to others. Perhaps just ask?

Which Data to Solicit. Some people, particularly advertisers, seek to provide personalization based on a small amount of information. For instance, recommendations might be based on demographic data (e.g. *ZAG* — zip code, age, gender), or generalized preferences of attributes describing the items involved (in movies, this might mean the user's favorite genres). Even this seemingly-innocuous amount of information can have a striking effect on one's privacy. Sweeney showed that information *similar* to ZAG may be highly identifying: 87% of the people in the 1990 U.S. census are likely to be uniquely identified based on only zip code, *birthdate*, and gender [10].

Highly personalized recommenders, such as those based on automated collaborative filtering, require a far higher degree of personal preference information from the user. These requirements lead to even larger privacy concerns since this level of preference information may reveal substantial personal information about the user.

In past work we explored eliciting information from new users in the MovieLens movie recommender system in VOI-aware ways that optimize both the required user effort and initial recommendation accuracy [13, 14]. We built interfaces that successfully reduced the user effort needed to start receiving recommendations. Moreover, we found that people who used the VOI-aware interfaces received more accurate recommendations than people who were did not use the enhanced interfaces.

In the interest of user privacy, this kind of approach may be comforting to some users in that fewer discrete pieces of information (e.g. movie ratings) need to be provided before the system becomes accurate. However, since the recommendations are better, quite possibly the user has given up a greater amount of information about herself than she would have with an unoptimized approach.

Selectively Discarding Data. If a recommender knows "too much" about a user, which data should be kept? The answer to this question is not necessarily the information with the highest value. If "low-valued" information is discarded from many users' models, then perhaps that information is no longer low-valued since it has become more rare. Choosing an appropriate set of data that balances both the benefit to the overall community and the quality of each individual user model seems challenging.

User Interface. What should the user interface look like, especially after the system thinks it has learned enough about the user? What if the user *wants* to tell us more about herself? How does one present a privacy-preserving recommender system in an understandable way? In our experiences with MovieLens, we have found no shortage of people willing to provide hundreds and sometimes thousands of movie ratings. Indeed, user feedback from our periodic surveys reveals that rating movies is among the leading reasons people have for using the system! These observations that some users want to give up their information may make it tricky to create a usable interface that effectively conveys the privacy-preserving aspects of the recommender system.

Impact on CF Algorithms. How well do current collaborative filtering algorithms operate in reduced-data environments? Many different aspects of recommendation quality might be affected: accuracy, coverage, novelty, and so on. There is some evidence that it is possible to hide or change ratings and still have good recommendations. Berkovsky et. al. looked at the performance of distributed recommender algorithms when obfuscating (hiding) the ratings of users in a 100% dense subset of the Jester dataset of jokes [9]. Polat et. al. looked at the performance of collaborative filtering algorithms when randomly perturbing rating values in the Jester and MovieLens 100K datasets [8]. In both cases, the recommendations did become less accurate, but it is unclear whether the drop is noticeable to users.

Further practical privacy-preserving algorithms and tests on other datasets would be valuable. In particular, the highly dense Jester dataset may not reflect the results of most recommender systems, because usually a recommender is used when users cannot possibly rate all items, hence the data is very sparse. Some algorithms such as SVD seem more naturally suited for sparse data sets [15] — are they even better if given selectively chosen high-information data? Is this data inherently less noisy, and if so, could it even lead to *better* recommendations using specialized algorithms?

2.2 Exposure Risk

In this section, we discuss the potential risks to users who divulge personal information. There is the direct risk that someone will learn information that the user wished to keep private. For example, revealing identifying information could lead to identity theft. There are also indirect risks of *re-identification* — finding information about a user in one system that could identify her in another system [10]. The user may not have expected others to be able to “connect” her identities from the two systems (e.g. a personal webpage and a controversial blog written under a pseudonym).

Combinations of attributes may be highly identifying. Such combinations are sometimes called a *quasi-identifier* to differentiate them from directly identifying information like social security number. Our earlier example showed the combination of 5-digit zip code, birthdate, and gender to be a quasi-identifier. This was used to identify former Massachusetts governor William Weld in voter registration records, and then re-identify him in a supposedly anonymous medical records dataset given to researchers [10]!

Personal preferences like those expressed to many recommender systems may also turn out to be a quasi-identifier, especially if people express unusual preferences. That may allow undesired re-identification using only preference data. Furthermore, a centralized recommender system might be able to re-identify its users in locations those users did not expect. For example, perhaps Amazon.com could find former customers on a competitor's site and offer incentives to lure them away from the competitor.

Whether identification or re-identification is unwelcome is likely to vary by domain and by user. In some domains, such as music, some users may be open to sharing their tastes with others. In other domains, such as medical information, users may have serious concerns about sharing their preferences with anyone, because of the potential harm should the information leak to colleagues, employers, or insurers. In still other domains, such as scientific research papers, the sensitivity of the information may vary with time. While working on a paper, a researcher may not want others to know what related work she is studying; however, once the paper is published, the list of references is publicly available and no longer presents a privacy concern.

3 Recommender Algorithm Security

Now, we turn to another violation of trust — recommendation bias. There are many ways to bias a recommender system. Here, we ignore "typical" computer attacks such as breaking in to and directly modifying the system, and instead focus on ones specific to the recommender algorithm. In particular, we examine a *shilling attack*, which attempts to manipulate the system's recommendations for a particular item by submitting misrepresented opinions to the system. We discuss the motivation for shilling (3.1), research on specific attack types (3.2), defending against attacks (3.3), and open questions about how system modifications for privacy might affect vulnerability to shilling (3.4).

3.1 Motivation for Shilling Attacks

One of the primary uses for a recommender system is to help people make decisions. Naturally, this makes recommender systems very interesting to people with vested interests in what people choose. For instance, a restaurant owner would be more successful if more people ate at his establishment, so it is within his best interests to have it recommended often. One way to do this is to provide good service to garner a good reputation among restaurant diners. This would lead to more frequent recommendation as users express high opinions of the restaurant.

A more underhanded and perhaps cheaper way to increase recommendation frequency is to manipulate the system into doing so by executing a shilling attack. Alternatively, an attack could be used to reduce the recommendation frequency for a competitor's offering. In either case, the attacker will profit as the manipulated recommendations cause more people to choose what he wants. As noted in section 1, this actually happens: a number of book reviews published on Amazon.com are written by the author of the book being reviewed.

3.2 Known Attack Variants

A shilling attack may be executed by having a group of users (human or agent) provide specially crafted "opinions" to a recommender system that cause it behave as desired. Different attacks specify different ways to construct the users' opinions and have varying degrees of success depending on the collaborative filtering algorithm used by the targeted recommender system. Each attack has a cost, measured by the amount of knowledge required to execute it and the amount of work that needs to be done (e.g. number of new identities or new ratings needed).

Our previous work [16] describes two very simple attacks *RandomBot* and *AverageBot* that can be carried out with a small amount of information about the user and item population. When executed against the k-Nearest-Neighbor algorithms commonly in use today, these attacks are indeed effective in changing a target item's recommendation frequency. Moreover, the attacks are non-trivial to detect with typical measures of recommender system performance.

More recently, Mobasher, et al., show that the basic attacks described in [16] can be improved with a modicum of additional information about users and items. In particular, they find that it is possible to target an attack to strongly affect recommendations for a specific *segment* of the user population [17]. This focused attack has a lower cost per unit effect than the RandomBot or AverageBot attacks, so they can be useful for adversaries who know what demographic of people they would like to target (i.e. targeted marketing campaigns) and who have limited resources to mount an attack with.

3.3 Defending Against Attacks

To formulate a response to shilling attacks, we examine a very similar attack faced by operators of reputation management and peer-to-peer systems, the *Sybil attack* [18]. In this type of attack, an attacker creates false identities that collude to achieve some objective such as increasing the reputation of an identity or increasing the influence of a node in a peer-to-peer network. For example, consider a dishonest seller on eBay who wishes to increase his feedback score. He could create a large number of identities and use them to leave himself positive feedback. This might increase the chances that a buyer will trust him and thus be tricked into purchasing an item from him.

Sybil attacks may be addressed by developing attack-resistant algorithms [19, 20], or increasing the cost of acquiring identities [21]. These ideas can be used to defend against shilling attacks as well. Work on shilling-resistant collaborative filtering algorithms is an active area of research. O'Donovan and Smyth show that an algorithm based on implicit trust scores that are computed from the accuracy of past recommendations can make shilling attacks less effective [22].

The other approach, making identities more expensive, is a simple-sounding solution that would prevent an attack from even reaching the recommender algorithm in the first place. However, the use of CAPTCHAs [23] or requiring some other non-trivial commitment of resources (e.g. monetary or computational) are believed to be either overly exclusive and unfair [21] or ineffective at preventing Sybil attacks due to unrealistic assumptions about users and attackers [18]. Thus, marginally increasing the cost of creating identities may be only a stopgap defense against shilling attacks.

There are more traditional cryptographic solutions of identity validation such as those described in [21] where the system uses a trusted third party to ensure that each person can only establish one identity. This can substantially raise the cost of an attack, but also raises privacy concerns as it requires that users reveal their identity to some entity just to use the system. Furthermore, if the trusted third party manages identities in multiple systems, it becomes possible to track one person across them, which increases the risk of re-identification.

3.4 Open Questions - Privacy and Shilling

The desire to preserve the privacy of users in a recommender system may confound the security problems. If we modify recommender systems to preserve user privacy, does that change how they are affected by shilling attacks? Likewise, as discussed above, defending against attacks may cause the loss of some privacy. What kinds of trade-offs between privacy and security might a recommender system operator need to make?

Attack Effectiveness. Do shilling attacks become more effective against privacy-preserving recommender systems? As additional privacy is introduced to a recommender system, the opportunities for attacks can increase considerably. Our work [16] shows that attacks that target recommendation frequency of low-information items (i.e. ones with few ratings) are more effective than attacks against high-information items.

In a system that tries to maintain a minimal amount of information about its members, it is possible that *every* item might have sufficiently few ratings to be vulnerable to highly-effective attacks.

Attack Difficulty. Are shilling attacks more or less difficult to mount against privacy-preserving recommender systems? As mentioned above, more individual items might become targets for effective attacks. On the other hand, if the recommender system only keeps a subset of data provided to it, an attack strategy will need to take that into consideration, both for the users being targeted and for the users introduced by the attack. This would require the attacker to know more about the system being attacked, thus increasing the cost of an attack.

Another possible impeding factor in an attack is the interface presented to users. A VOI-aware interface such as the ones used in our past work [13, 14] can control which items may be rated by a user in order to maximize the information gain per collected rating. This significantly constrains what an attacker can do and could make it more difficult to impact the system in precise ways.

Attack Detection. How does one detect shilling attacks? There are ways of detecting automated agents that are not specific to recommenders, such as noting patterns in account names, or the source or speed of account creation or opinion submission. Are there also robust ways of detecting profiles that differ significantly from normal, or that clearly affect particular items in unusual ways?

Moreover, in a privacy-preserving recommender system, is it easier or harder to detect an attack? One might theorize that in a low-data environment, it becomes easier to identify atypical patterns that are indicative of an attack. If true, this would certainly be a boon to recommender system operators. On the other hand, discarding some of the data entered by a shilling agent might leave the remaining data looking more like a human, and hence harder to detect.

4 Distributed Recommenders

4.1 Motivation

Users of MovieLens write to thank us for running a non-commercial recommender. They feel they can trust our recommendations because we do not have an external motivation to push them towards one movie or away from another.

Because traditional recommenders require large centralized resources, they must be run by some organization. That organization has control of the recommender cloud in figure 1: the data and algorithms used to form the recommendations, and even the user interface through which recommendations are presented. There are several reasons that users might wish to have more control over the recommendations. First, users might fear the centralized organization will expose their personal information. They might prefer to control their own data. Second, users might be concerned that the recommendations provided will be biased for the good of the organization rather than their own good. They might wish to have some assurances about the recommendation algorithm being used. They might even prefer to be able to select the recommendation algorithms by themselves, rather than have those algorithms chosen by someone else.

The high-level research question in this section is: can recommender systems be developed in which there is no centralized authority that can co-opt the recommendation process? A positive answer to this question might be based on developing a recommendation algorithm that has no centralized authority, limiting what the centralized authority can do, or verifying that the centralized authority is meeting certain standards of behavior in its actions. The first two approaches have been investigated in past research.

4.2 Prior Approaches

One such approach enables a community to build a shared view of a recommendation model, even though individuals only share cryptographically protected versions of their ratings vectors (preferences). Canny described a recommender system in which a centralized singular value decomposition model is built by a *tallier* combining encrypted ratings vectors from each user [24]. For security, there might be multiple, distributed talliers; indeed, each client might also be a tallier. Attackers cannot learn the original ratings vectors from the encrypted ratings vectors, but users can check that their uncorrupted ratings data is used in the model using a zero knowledge proof technique.

This approach protects against exposure of personal information, since no one can see the original ratings vectors. Canny also shows that the model-building algorithm protects against the model being unnoticeably corrupted if at least half the talliers are honest. Note that this does not protect against all forms of bias. For example, clients can still shill by providing false preferences in the correct protocol that is then dutifully incorporated into the model by talliers.

Miller et al. extends Canny's work by using Canny's approach to computation, but with an item-item recommendation algorithm [2, 25]. The idea is the same: encrypted ratings vectors are distributed by users; the vectors cannot be reverse-engineered to produce the original ratings; and a centralized model is built that can be used to produce individual recommendations [26]. The individual recommendations can be produced by a user by combining their own ratings with the model without sharing those ratings with anyone else.

One key advantage of Miller's algorithm is that it can produce models incrementally by collecting ratings vectors over time. In principle, each user could keep his own model, only sharing encrypted ratings data with others. Such a user might be satisfied with a partial model that was only suitable for making recommendations for himself, not for other users. Miller showed that these models are small, fast, and could easily be maintained on a personal workstation. Ratings could be distributed using a variety of well-known peer-to-peer approaches, such as those used in Gnutella[5], Freenet [27], or Chord [28].

In the extreme, the smaller model could be maintained on a mobile device. Distributing ratings to these devices would be more challenging, since they are only occasionally connected to the Internet. One radical idea is that the ratings might be distributed wirelessly using a personal-area network like Bluetooth. In this vision, the user walks around the world carrying her mobile device, which shares encrypted ratings vectors with nearby mobile devices. The encryption of the ratings vectors would protect privacy, while the resulting distributed recommendation model would provide accurate recommendations using a recommendation algorithm the user chose and maintained herself.

4.3 Open Questions

There are many open questions about the use of distributed recommenders that protect privacy or give individual control over the use of the ratings or recommender model. This section outlines some of the most important.

Practical Distributed Recommenders. Do distributed recommenders really work in practice? Do they lead to recommendations that are as accurate as those predicted by the analysis and offline experiments that have been performed? Actually implementing a distributed recommender system for a large user community, such as music or movie lovers, and solving the practical problems faced by such a system would be a substantial research contribution.

[5] http://rfc-gnutella.sourceforge.net/

An interesting area for experimentation is to investigate what would really happen with the distribution of ratings data over personal area networks such as Bluetooth. Would users be exposed to enough different types of people to get a wide variety of recommendations, or would there be too much similarity in the people they encounter on a day-to-day basis?

Integrity. Security attacks are especially of concern for distributed recommenders, because their ratings vectors would likely be shared openly through well-known protocols. (In principle the ratings vectors could be shared through a secure channel, but then only certified programs could participate in the recommendation process, a result that would be less satisfying to the peer-to-peer community, for example.) These ratings vectors could be destroyed or discarded as they are passed through the system. More simply, shilling attacks from robot “users” could be injected into the system as described in section 3. Since a distributed system makes it difficult to verify identity, these attacks would be challenging to thwart. What mechanisms could be developed to make shilling attacks more difficult in a distributed recommender system?

The bottom-line goal of the research questions in this section is to develop recommenders that are guaranteed to serve the needs of their end-users. What techniques other than those discussed here could provide such guarantees? Could certification techniques show with high probability that the recommendations are made honestly? Are there zero-knowledge proofs that show not only that the data used is the correct data, but also that the algorithms used have the desired properties? Research that could demonstrate properties of centralized recommender algorithms might be invaluable.

5 Conclusion

The issues of privacy and security in recommender systems is rich with important, unanswered research questions. Highly personalized recommender systems, like those discussed in this paper, collect large volumes of very personal data about their users. How can security techniques be used to guarantee that this personal data will never be leaked without the permission of its subject? Further, these recommender systems are increasingly important in guiding people’s decisions about what they want to do, what they want to buy, even where they want to go. How can the users be sure that the recommendations they receive have not been inappropriately influenced or modified?

In this paper we explored three aspects of recommender systems that relate to these privacy and security questions: value and risks of personal information, shilling, and distributed recommenders. Previous work on value of information (VOI) shows that it can be used to more effectively collect information from new users. We believe it can similarly be used to determine when to stop collecting information to properly balance the privacy given up by users with the quality of the recommendations, and to intelligently choose which information to discard if “too much” is known about a user. The challenge of shilling is that the aforementioned privacy protections may make shilling easier, especially if they reduce the amount of information the recommender system keeps about each user. Past research in distributed recommenders has shown that security techniques such as cryptosystems and zero knowledge proofs can be used to provide recommenders with dramatically different security and privacy properties.

Rather than try to completely define the set of privacy and security issues involving recommenders, we have tried to outline some of the most important issues, and to identify some key research questions that may yield to the research techniques of the security community. We hope by raising these questions to inspire even more high quality research into the security and privacy implications of the increasingly important ubiquity of recommender systems.

Acknowledgments

Thanks to the participants of the 2005 User Modeling Workshop on Privacy-Enhanced Personalization for their helpful suggestions. Thanks also to members of GroupLens Research at the University of Minnesota for many exciting research collaborations, fruitful discussions, and overall fun. Particular thanks are due to our colleagues Al Mamunur Rashid, Istvan Albert, Dan Cosley, Sean M. McNee, and Joseph Konstan, our co-authors on the VOI research [13, 14], and to our colleagues Joseph Konstan and Brad Miller, our co-authors on the PocketLens research [26]. This work was supported by grants from the NSF (DGE 95-54517, IIS 96-13960, IIS 97-34442, IIS 99-78717, and IIS 01-02229).

References

1. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An open architecture for collaborative filtering of netnews. In: CSCW '94: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, Chapel Hill, North Carolina, United States, ACM Press (1994) 175–186
2. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: WWW '01: Proceedings of the 10th International Conference on World Wide Web, Hong Kong, ACM Press (2001) 285–295
3. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering (2005) 734–749
4. Ackerman, M.S., Cranor, L.F., Reagle, J.: Privacy in e-commerce: Examining user scenarios and privacy preferences. In: ACM Conference on Electronic Commerce. (1999) 1–8
5. Ali, K., van Stam, W.: TiVo: Making show recommendations using a distributed collaborative filtering architecture. In: KDD '04: Knowledge Discovery and Data Mining Conference, Seattle, Washington, USA (2004) 394 – 401
6. Ramakrishnan, N., Keller, B.J., Mirza, B.J., Grama, A., Karypis, G.: Privacy risks in recommender systems. IEEE Internet Computing **5** (2001) 54–62
7. Lam, S.K., Riedl, J.: Privacy, shilling, and the value of information in recommender systems. In: Proceedings of User Modeling Workshop on Privacy-Enhanced Personalization. (2005) 85–92

8. Polat, H., Du, W.: Privacy-preserving collaborative filtering using randomized perturbation techniques. In: ICDM ’03: Proceedings of the Third IEEE International Conference on Data Mining. (2003)
9. Berkovsky, S., Eytani, Y., Kuflik, T., Ricci, F.: Privacy-enhanced collaborative filtering. In: Proceedings of User Modeling Workshop on Privacy-Enhanced Personalization. (2005) 75–83
10. Sweeney, L.: k-Anonymity: A model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems (2002) 557–570
11. Verykios, V.S., Bertino, E., Fovino, I.N., Provenza, L.P., Aygin, Y., Theodoridis, Y.: State-of-the-art in privacy preserving data mining. In: SIGMOD ’05: Proceedings of the Conference on the Management of Data. (2005)
12. Pennock, D.M., Horvitz, E., Lawrence, S., Giles, C.L.: Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach. In: UAI ’00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, Stanford, CA, Morgan Kaufmann Publishers Inc. (2000) 473–480
13. Rashid, A.M., Albert, I., Cosley, D., Lam, S.K., McNee, S., Konstan, J.A., Riedl, J.: Getting to know you: Learning new user preferences in recommender systems. In: Proceedings of the 2002 International Conference on Intelligent User Interfaces, San Francisco, CA (2002) 127–134
14. McNee, S.M., Lam, S.K., Konstan, J.A., Riedl, J.: Interfaces for eliciting new user preferences in recommender systems. In: User Modeling, Johnstown, PA, USA, Springer Verlag (2003) 178–187
15. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Application of dimensionality reduction in recommender system – a case study. In: ACM WebKDD 2000 Web Mining for E-Commerce Workshop, Boston, MA, USA (2000)
16. Lam, S.K., Riedl, J.: Shilling recommender systems for fun and profit. In: WWW ’04: Proceedings of the 13th International Conference on World Wide Web, New York, NY, USA, ACM Press (2004) 393–402
17. Burke, R., Mobasher, B., Zabicki, R., Bhaumik, R.: Identifying attack models for secure recommendation. In: ACM IUI Workshop: Beyond Personalization. (2005)
18. Douceur, J.: The Sybil attack. In: Proceedings of the 1st International Workshop on Peer-to-Peer Systems. (2002)
19. Dellarocas, C.: Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In: ACM Conference on Electronic Commerce. (2000) 150–157
20. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The Eigentrust algorithm for reputation management in P2P networks. In: WWW ’03: Proceedings of the 12th International Conference on World Wide Web, New York, NY, USA, ACM Press (2003) 640–651
21. Friedman, E., Resnick, P.: The social cost of cheap pseudonyms. Journal of Economics and Management Strategy (1999)
22. O’Donovan, J., Smyth, B.: Is trust robust?: An analysis of trust-based recommendation. In: IUI ’06: Proceedings of the 11th International Conference on Intelligent User Interfaces, New York, NY, USA, ACM Press (2006) 101–108
23. von Ahn, L., Blum, M., Hopper, N., Langford, J.: CAPTCHA: Using hard AI problems for security. In: Proceedings of Eurocrypt, 2003. (2003)
24. Canny, J.: Collaborative filtering with privacy via factor analysis. In: SIGIR ’02: Proceedings of the 25th International ACM Conference on Research and Development in Information Retrieval, Tampere, Finland, ACM Press (2002) 238–245

25. Karypis, G.: Evaluation of item-based top-n recommendation algorithms. In: Proceedings of the 10th Conference of Information and Knowledge Management. (2001)
26. Miller, B.N., Konstan, J.A., Riedl, J.: Pocketlens: Toward a personal recommender system. ACM Transactions on Information Systems **22** (2004) 437–476
27. Clarke, I., Hong, T.W., Miller, S.G., Sandberg, O., Wiley, B.: Protecting free expression online with Freenet. IEEE Internet Computing (2002)
28. Stoica, I., Morris, R., Karger, D., Kaashoek, F., Balakrishnan, H.: Chord: A scalable Peer-To-Peer lookup service for internet applications. In: Proceedings of the 2001 ACM SIGCOMM Conference. (2001) 149–160