Using Recommenders for Discretionary Access Control

Suresh Chari Larry Koved Mary Ellen Zurko
IBM Research IBM Software Group
Hawthorne,NY Westford, MA
Email: {schari,koved,mzurko}@us.ibm.com

Abstract

Enterprises increasingly subscribe to Software as a Service (SaaS) applications for collaboration. In the past, enterprise organizational boundaries have been key to the controls on sharing, providing both a social and technical boundary that can slow or stop potentially inappropriate sharing. However, SaaS collaboration technology is often used to across enterprise boundaries. One of the key security concerns with this migration of in-enterprise, on-premises applications to application clouds is that of information flow across enterprise boundaries and specifically how this may be monitored and controlled. Business use of collaboration systems requires end-users to rapidly make a multitude of discretionary access control decisions. Many of the features that lower friction in collaboration can encourage user slip ups, and traditional notions of access control, such as RBAC are not very usable by people without information security experience, nor is it amenable for low overhead business collaborations. This paper proposes using recommender systems for proposing discretionary access control policies. Specifically, to notice similar patterns of use for similar content and to suggest access control policies derived from context and prior communication patterns based on past interactions.

1 Introduction

Enterprises increasingly subscribe to Software as a Service (SaaS) applications for collaboration. Such collaboration systems allow users to share documents, chat, mail, calendar entries, and videos, and so on with other end-users in the system. Enterprise organizational boundaries have in the past been key to the controls on sharing, providing both a social and technical boundary that can slow or stop potentially inappropriate sharing. Key motivating use cases for SaaS collaboration are when work needs to occur across enterprise boundaries. In these cases, collaborators may be part of the same business organization or completely different organizations. One of the key security concerns with this migration of in-enterprise, on-premises applications to application clouds is that of information flow across enterprise boundaries and specifically how this may be monitored and controlled. Whereas an enterprise may choose to deploy an organization defined mandatory access control policy at its boundaries, through the use of Data Leak Prevention (DLP) technologies, defining inter-organization SaaS application access control policies has been more challenging.

Business use of collaboration systems requires end-users to rapidly make a multitude of discretionary access control decisions about whom they are willing to share documents, whom not to share with, whom to invite for an online web-meeting, and so on. Many of the features that lower friction in collaboration can encourage user slip ups, and there are many examples of information leakage through end-users making incorrect content sharing decisions (see for example[4]). Traditional notions of access control, such as RBAC [5], are not very usable by people without information security experience [10]. Also, structured access control such as RBAC does not map to low overhead business collaboration. As a result, in the collaboration and social networking contexts, end-users themselves are responsible for defining the security policies to objects which they have created and intend to share with a (potentially) limited group of colleagues and collaborators. This typically leads users to make default choices which results in poor security (e.g., [4, 6]).

Our solution proposes to leverage recommender systems (see for example [1]) to address the challenge of defining discretionary access control issues. We have chosen this direction for a number of reasons. One reason is our desire to explore how collaboration mechanisms can bring additional usability to security. The purpose of recommender systems is to notice similar patterns of use for similar content, and to propose suggestions based on these similarities. We posit that early on in a business collaboration, initial

sharing is done with more care and attention than later sharing. We also presume that existing and established patterns of sharing are honoring the desired enterprise controls. This second assumption needs to be validated. For example, we would not want the system to actually automate implementing any recommendations, as one temporary bad pattern could then have cascading negative effects . Thus, to address the challenge of defining discretionary access control in collaborative systems, we are designing a system where security policies are defined and enforced in the context of each end-user task, i.e. to whom an email is sent, or with whom to share a particular document. This is, in particular, determined by the specifics of the content that is being shared in the given context. In this position paper, we briefly outline a solution that we have been exploring in the context of a collaboration application.

2 Models

In our system model we have end-users of an organization who interact with our collaboration system via multiple channels. We choose organizations as a primitive of our model to align with business collaboration systems such as LotusLive [7]. Commonly used channels include web user agents, such as web browsers and mobile browsers. They may also be collaborative containers of information, such as files, instant messages, and meetings. Access control policies for document sharing can be dependent the end-user sharing, the end-user being shared with, the organization of each (all) end-users involved, and the channel through which the end-user interacts with the collaboration system. For instance, organizations may disallow sharing of sensitive documents with users with access over a mobile channel. We assume that end-users belong to one or more social-networks supported by the collaboration application and are members of multiple sub-networks in these social-networks. Social networks may be restricted to a single organization, or may cross organizational boundaries. They are assumed to align with organizational boundaries, policies, and business goals. It is recognized that some informal inter-personal sharing will also occur in this context, and it is a challenge of this work to ensure that recommendations are business appropriate.

The data to which we are trying to attach access control policies in this system are typically documents, mail, calendar entries, instant messages, video or documents shared in web meetings. We assume that, along with the raw content, we have access to classifiers which can categorize content and produce meta-data, such as the classification tags to be associated with the raw content (see for example [11]). We assume that we have appropriate classifiers for each content type (text, audio, video, etc.) and that for each piece of content we have enough meta-data so we can apply the heuristics for recommendation engines as described in the next section. In certain contexts this may be an overly optimistic assumption. For instance: if the context is an instant message such as *"hi"*, there is very little classification possible for the content in this setting solely based on the message.

A more robust classification system may need to attempt to correlate content across multiple communication containers and channels. Given this informal model, the problem we are trying to address is: Given a context (some data, one or more channels of interaction, and one or more social networks) the recommender system should output a ordered list of names of other end-users in the systems such that

1. Sharing the document in that channel with anyone on the list doesn't violate the end-user's organization mandatory access control policies (if one exists).

2. The ordered list approximates as closely as possible the end-user's desired set of people with whom rhe content may be shared.

We can set up simple metrics by which to evaluate the recommender system by comparing the recommended list with the actual set of people with whom the end-user shares the document. As with other learning systems we assume that the system learns user behavior over time. The formal evaluation of our approach will be done after the system learns user behavior on a number of initial data points.

3 Recommendation Engine Heuristics

We are in the process of experimenting with a number of heuristic techniques for the recommendation engines and evaluating these against test data. In the current phase, we are foremost interested in building a set of heuristics which yield good results for accurate recommendations, and we discount the latency inherent in some of the more detailed heuristics. The general problem can be seen as a variant of the *expert-search problem*: Given a training set which consists of a list of experts for query terms, given a query we wish to find an ordered list of experts for this query. There are many heuristics proposed for this

problem[2]. In related work these techniques have been used to suggest recipients for emails [9]. Here we sketch the details of some of the heuristics we are experimenting with.

- *Using decisions from similar documents*: The first and perhaps most important heuristic is to induct from what this user did to similar documents in the same channel. For example, from the training data we look at documents which are similar in the sense of having a number of overlapping classification tags and look to see with whom those documents were shared. We give a higher weight to documents which have a higher degree of similarity and then among documents with equal degree of similarity we want to give a higher preference to those whom the end-user has shared a larger number of similar documents. The k-Nearest Neighbor heuristic has been shown to be a particularly effective heuristic in the expert search problem [2] and this is the primary heuristic we use. Assume that we have a metric, $sim(d, d_1)$ which measures the similarity between two documents d and d_1. We can define a measure which is proportional to our estimate of the probability that the current document d is shared with a given end-user s as given by the formula:

  $$P(s, d) = \sum_{d_1 \in \mathcal{K}} (sim(d, d_1) * shared(s, d_1))$$

  where $\mathcal{K}$ is the set of the k documents most similar to the document d as given by the similarity metric $sim()$ and $shared(s, d_1)$ is 1 if d_1 was shared with user s and 0 otherwise. There is a lot of work on what the correct notions of document similarity are and we can choose a commonly accepted notion of cosine similarity [3]. There are many other variations of this notion of similarity and we intend to evaluate which ones work well in practice. For text documents we can directly use the *similar document* searches offered by packages such as Lucene[8]. With appropriate content classifiers we can do this similarly to other content types.

  While the highest priority is for history of sharing of documents along the same channel we want to consider how similar documents have been shared on other channels.

- *Negative recommendations*: Mandatory access control rules such as data leakage prevention rules, as well as more sophisticated separation of duty type rules, can be viewed in this framework as making negative recommendations on sharing. In our system we will remove any user who appears in the positive recommendation lists sharing with whom will result in a violation of any mandatory access control policies. If the user insists on sharing with such users, there are a couple of obvious choices. The first is to disallow sharing. The other choice is to mark this as a violation and log this as an exception as part of an auditing process. Simplistically, we view negative recommendations as a post processing step after other recommenders have run. This is sufficient for a number of policies such as monitoring the flow of confidential information across corporate boundaries. In the use-case scenarios we have described, negative recommendations are always binding. There are cases where negative recommendation engines can be seen as attaching a risk to the intended sharer. This can be incorporated in the mathematical model above as a negative term. In future work we will examine this idea further.

- *Inducting from the broader social network*: While we expect that sharing decisions on similar documents will most likely be the best predictor for the current document, we expect that sharing behavior in larger social networks can also be used to make recommendations. Given a particular topic, we can expect that multiple people from the users social network may be generating documents with the same classification. Thus it may be natural to extend this idea of document similarity to include as corpus all documents generated by the end-users' entire social network. Thus, we are using a collaborative filtering like approach: We recommend that this user share this content with the same set of people with whom the larger social network shared similar content. Technical implementation of this is essentially the same as before since we are just varying the corpus on which we judge similarity. There will, however, create some interesting challenges with respect to privacy given that this could result in information leakage about topics that are not intended to be shared outside a particular social network.

4 Techniques for evaluation

Currently, we plan to evaluate these recommendation engines based on an offline analysis of a corpus of data of a content management system. This data is split into a training set which is what we base our

recommendations on and the test set is what we will use to evaluate the success. In a preprocessing step, the documents which are being shared in both the test set and the training set are run through different classifiers to obtain the tags for classification. We use these tags as the basis for similarity comparisons of the documents. In the current version, to focus the evaluation on the statistical techniques, we assume that all the sharing is on the same channel. We are in the midst of this evaluation phase and will release results as they are available. After we apply training on a set of test documents, we would like to use ongoing use of the system to provide feedback to the system for learning and improving the accuracy of the access control recommendations. Each set of prompts to suggest with whom to share d, is an opportunity to get feedback to use for training. We envision options such as share, dont share, never share, etc., are the input to the next training cycle. There are other possible inputs into the training, such as order of the To, Cc, and Bcc lists.

5 Conclusion

The central thesis of this position paper is that access control systems, such as RBAC, is often not useful for end-users in collaboration systems. As such, we need new ways for end-users to describe security properties. However, studies of end-users has shown that they frequently do not understand the security implications of the security settings that they choose. From this perspective, we propose that we look at security from the perspective of providing a set of value add functions from which we can derive appropriate security attributes. In this position paper, we have proposed that we use recommender systems to suggest discretionary access control policies. While nominally offered as a value-add feature, we use the results of the user interaction to derive the access control properties in collaboration systems. As is typical in Web 2.0 collaboration systems, we propose to learning new discretionary access control policies by building up a knowledge base through the aggregation of results over a large number of interactions with the system. The more documents that are created and classified / recommended, the better the system will be at deriving proposed access control policies. Whereas many corporate content sharing systems would like to define and support mandatory access control policies, such as data leakage prevention (DLP), we propose to support these types of policies through simple feedback driven learning mechanisms. We are in the process of collecting data to train the system and evaluate the effectiveness of the approach proposed in this position paper. From experience in building security for collaboration systems, we recognize the need to make the security of these systems easier to use.

References

[1] S.Perugini, M. Goncalves and E. Fox. Recommender Systems Research: A Connection-Centric Survey. Journal of Intelligent Information Systems, Vol. 23 , Issue 2, pp 107 – 143, 2004.

[2] K. Balog, L. Azzopardi and M. de Rijke. Formal models for expert finding in enterprise corpora. SIGIR 2006: Proceedings of 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

[3] Y. Yang and X. Liu A re-examination of text categorization methods SIGIR 1999: Process of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

[4] Nathaniel S. Good and Aaron Krekelberg Usability and privacy: a study of Kazaa P2P file-sharing Proceedings of the SIGCHI conference on Human factors in computing systems Ft. Lauderdale, Florida, USA. pp 137 – 144, 2003.

[5] D.F. Ferraiolo and D.R. Kuhn Role Based Access Control Proceeding of the 15th National Computer Security Conference, Oct 13-16, 1992, pp. 554-563.

[6] M. Zurko, C. Kaufman, K. Spanbauer and C.Bassett. Did You Ever Have to Make Up Your Mind? What Notes Users Do When Faced With a Security Decision. Proceedings of Annual Computer Security Applications Conference (ACSAC), December 2002.

[7] The Lotus Live Collaboration Platform https://www.lotuslive.com/

[8] The Apache Lucene Project. http://lucene.apache.org/.

[9] Vitor Carvalho and William Cohen. Ranking Users for Intelligent Message Addressing. Proceedings of the 30th European Conference on Information Retrieval, Glasgow, England, 30th March 3rd April, 2008.

[10] M. Zurko and R. Simon and T. Sanfilippo A User-Centered, Modular Authorization Service Built on an RBAC Foundation. Proceedings of the IEEE Symposium on Security and Privacy 1999, pp 57-71.

[11] The Unstructured Information Management Applications project. http://incubator.apache.org/uima/.