How Users Use Access Control

D. K. Smetters
PARC
3333 Coyote Hill Rd
Palo Alto, CA, USA
smetters@parc.com

Nathan Good
PARC
3333 Coyote Hill Rd
Palo Alto, CA, USA
ngood@parc.com

ABSTRACT

Existing technologies for file sharing differ widely in the granularity of control they give users over who can access their data; achieving finer-grained control generally requires more user effort. We want to understand what level of control users need over their data, by examining what sorts of access policies users actually create in practice.

We used automated data mining techniques to examine the real-world use of access control features present in standard document sharing systems in a corporate environment as used over a long (> 10 year) time span. We find that while users rarely need to change access policies, the policies they do express are actually quite complex. We also find that users participate in larger numbers of access control and email sharing groups than measured by self-report in previous studies. We hypothesize that much of this complexity might be reduced by considering these policies as examples of simpler *access control patterns*. From our analysis of what access control features are used and where errors are made, we propose a set of design guidelines for access control systems themselves and the tools used to manage them, intended to increase usability and decrease error.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection—*access controls*; H.1.2 [**Models and Principles**]: User/Machine Systems—*human factors, human information processing*

General Terms

Design, Security, Human Factors

Keywords

usability, access control, file sharing

1. INTRODUCTION

In order to share files or other resources securely, users must implicitly or explicitly express a notion of *policy* – namely who should be able to access the shared content, and who should not. Existing technologies for sharing content differ widely in the granularity of control they offer to users, and the corresponding level of effort required to achieve that control.

For example, email offers a low-effort "fire and forget" form of *implicit*, coarse-grained policy specification. Only those on the `To:` or `cc:` list of a message are assumed to be able to read it and its attachments; and they cannot alter the sender's copy of those documents – *i.e.,* they have no write access. Access rights cannot be removed once granted, only expanded by sending the same documents to additional recipients.

Traditional document sharing and file systems allow *explicit* policy specifications, such as access control lists (ACLs), and offer fine granularity of resource control. They vary in the types of operations which can be controlled – from a simple separation between permission to read and to write a document, to separate control of deletion, search, and the management of access permissions themselves. Some allow for access to be explicitly denied or permitted, resulting in complex access lists requiring sophisticated evaluation rules to determine their effect. This control comes at the cost of potentially high user effort [19], tendency to error [10, 13] and the inability to control sharing outside one's administrative domain. Interestingly, even Microsoft's suggestions for "best practices" for managing permissions suggests that a number of available access control features should not be used to avoid error [3].

"Web 2.0" user-centered content sharing systems offer attempts at simplified security and privacy controls that are in between these alternatives. Aiming for ease of use, they offer users only limited means to specify with whom they will share – *e.g.,* no one ("private"), the world("public"), or a small number of user-definable lists of sharing partners (most commonly labeled "friends" and "family"). However, studies suggest they end up offering less control than users actually require [1, 11].

To enable easy, secure sharing in multiple contexts we need to find the appropriate balance between control and complexity. To achieve that, we need to understand what users' security needs actually are, and what effort they are willing to go to to achieve them. We expect that users may only make use of a subset of these systems available functionality. These systems may also not map well onto users' actual tasks. In response, they may try to use the existing features to approximate (successfully or unsuccessfully) the functionality they would prefer. Knowing what features people currently use, and how they use and abuse them, may allow us to design new features and interfaces better suited to their real tasks. Previous ethnographic studies of file and resource sharing [17, 16] have focused on asking users themselves to report on how they share and protect data. We were interested in collecting behavioral data over time,as users' self-descriptions of their own behavior can

Symposium on Usable Privacy and Security (SOUPS) 2009, July 15–17, 2009, Mountain View, CA USA
.

be incomplete or inaccurate [6].

We therefore turn to *cyberethnography* – analyzing the digital record of actual user behavior to better understand how a technology is used. In this case, we use automated data mining to examine how users in a medium-sized corporation make use of two very common access control features: the definition of access control groups, and the permissions settings, or ACLs, that users set on folders and documents. We analyzed this data in terms of the "work" done by users – examining when, and how, they actively set permissions on content, and what sorts of access control structures (*e.g.,* groups) they create and use to support that work. Our assumption is that it is always simpler for a user to ignore the question of access control entirely – to allow the content they create to fall under the control of whatever default access policy the system provides or had been previously set. When a user goes to the trouble to actually change that policy, it is interesting.

To understand how much flexibility users may want in sharing content, we compared the properties of four types of access control groups used concurrently over the course of 10 years to manage content in a single user community: Microsoft Windows® and Unix® user groups, email mailing lists and access control groups created in a commercial enterprise content management system (CMS), Xerox DocuShare®. CMS systems are designed for flexible sharing of documents and data among a potentially large number of collaborators, with a more intuitive user interface than traditional networked file systems. Windows and Unix user groups in this community were managed by administrators on users' and their own behalf, while users could define and manage DocuShare user groups and email mailing lists on their own.

We find that users set permissions on only a small minority of documents and folders. They rarely alter that content or its permissions after it is initially shared. But when they do specify access rights over a document or tree of documents, those specifications can be quite complex, granting rights to a number of groups and individuals. When users are given direct control over the creation and management of access control or sharing groups, they define a much larger number and variety of such groups than when that definition is managed by administrators. Both access control specifications and group definitions, however, are prone to inefficiencies and errors that can lead to lapses in security. We conclude that users do want a fair amount of expressive flexibility in the access control mechanisms they use, but they need additional support, perhaps through improved tools, to more effectively manage those policies.

We begin the remainder of the paper with a summary of related work. We then describe our experimental methods together with supporting background in section 3. We describe detailed results of our study and their analysis in section 4. In section 5 we conclude and discuss implications for design.

2. RELATED WORK

We divide related work into studies which look, in even a limited fashion, at the use of access control and sharing technologies in practice, and those which propose improved access control interfaces and evaluate them in a laboratory setting.

A number of recent studies have examined the use of access controls in file and media sharing applications. Voida *et.al.* [16] and Whalen *et.al.* [17] both looked at resource sharing within a corporate environment, using surveys and interviews to determine users' experiences with a variety of sharing technologies. Voida *et.al.* created a taxonomy of file sharing technologies available to their subjects, and categorized them on a number of dimensions, focusing particularly on requirements for initiation of sharing ("push" vs. "pull") and on facilities for providing notice of available information. As an indication of users' own estimates of the complexity of the sharing problem, Voida *et.al.*'s subjects report sharing documents with an average of only 7 individuals or groups. The study results of Whalen *et.al.* indicate that the majority of users (67% of their sample) do manage access control settings, and that they do so repeatedly over time.

Ahern *et.al.* studied the use of privacy settings in a mobile photo sharing application [1]. Participants uploaded photos taken from their cell phones to a photo sharing service [5], together with information about where they were taken; they could specify privacy settings for the photos at time of upload and also modify them later. Participant behavior was assessed through automated collection of data about photo upload and privacy settings, together with follow-up interviews. Ahern *et.al.* found that not only were the simple access controls available in this system too coarse-grained that users needs, but that users worked around them in order to achieve their sharing goals [1], often *oversharing*, or granting more access than they desired, in the process.

Lam and Churchill [8] examined how users assigned access controls over a large photo sharing website called Flickr. They found that only 20% of users had some form of access control on their photo collections, and the overwhelming majority of users did not change photos from the default setting of public (60% shared their entire photo collection). They did find that personalized groups or contact lists were used as a form of access control. Our study complements this work by providing quantitative explanations of access control settings within the enterprise, as well as an analysis of groups based access controls in the enterprise.

Bauer *et. al.* studied the access control policies users developed around physical keys and a more flexible replacement using mobile phones to control access to doors. Users found the automated alternative preferable to physical keys, and used it to manage more complex access policies.

All of these studies determined that their subjects had complex sharing needs that evolved over time, and that were inadequately supported by the access control technologies available to them.

To add further complexity, two studies demonstrate that requirements for control over shared resources may vary significantly between individuals, and as a function of the type of data or with whom it is being shared. Olson *et.al.* surveyed subjects about their willingness to share varying types of information, and identified categories of information and recipients subject to similar policies. Interestingly, for some categories the specific sharing policies applied varied between individuals, where others did not [12]. Miller and Edwards [11] interviewed users about their privacy and access policies around photo sharing, and found that users fell naturally into two groups. Privacy was a significant factor for the first, who expressed surprisingly complex (verbal) policies about what types of photos they would be comfortable sharing with various groups of recipients. The second group was largely unconcerned with privacy, and was comfortable making all of their photos public.

There is a long history of work on increasing the usability of interfaces to traditional access control systems, beginning with the work of Zurko *et. al.* [19, 18]. Permissions errors are particularly frustrating for users, as they are often only discovered at the time access is really needed, where they may not be easy to repair. While research prototypes have proposed mechanisms where a user setting an access control policy can assess its impact at the time of definition by visualizing access from the point of view of the intended recipient; such functionality is not available for common use. Several recent studies [10, 2, 13], have proposed alternative interfaces to traditional file system access control management. All of the proposed interfaces showed significant improvement in

user performance over the standard Windows access management interface; and in fact in these laboratory studies users attempting to set access permissions using the standard interface made significant numbers of errors.

The interface developed by Reeder *et. al.* in [13] is particularly interesting, not just because it presents a friendlier, more effective interface to controlling the complex Windows file system permissions, but that in order to make their system easy to use, they had to change the way those permissions operate, having the last change made by the user take precedence over all conflicting access specifications, including explicit denials of access. Their results strongly suggest that not only interfaces, but access control systems themselves must be tailored to achieve maximal usability.

3. BACKGROUND AND METHODS

We conducted our automated survey of access control use in a medium-sized corporation with a relatively stable population of approximately 200 employees. The systems we studied have all been in operation for at least 10 years. Our study protocol was approved by our institution's human subjects committee.

3.1 Access Groups

Most access control systems allow rights, or permissions, to be granted to not only individuals but also to named *groups* – sets of users and potentially other groups. This allows for a level of indirection and delegation in policy specification. For example, an administrator could create a group called *Human Resources* containing the current members of the Human Resources (HR) department; a user can then allow that group to access, say her insurance forms. She doesn't have to correctly remember the identity of all the users in the HR department, or update the permissions settings on her insurance forms when there is an HR personnel change. Such indirection, specifying all access in terms of named groups or *roles* is the foundation of Role-Based Access Control (RBAC) [4, 15]. Creation of such a group suggests that information will be repeatedly shared with that particular set of individuals, or at least repeatedly enough to offset the effort required for group creation. At the same time, specifying policies in terms of groups may cause problems through a lack of *transparency* – a user unaware of the actual membership of a group may accidentally share content in unintended ways. Access control systems differ in the manner in which they specify groups, whether groups can contain other groups as well as users, and in whether end users as well as administrators can manage group membership. Group membership information is normally visible to all members of an organization. This information must be public if users are to be able to determine what access control settings to apply in order to generate a particular access policy.

Group Data Collection We collected information about the number, name and membership of four types of access control groups (see Section 1). Unix and Windows group information was collected using simple shell and `awk` scripts. Email mailing lists were defined using a web interface, and were retrieved in machine readable form using a crawler program. DocuShare group information was collected using a modified form of the crawler program described in section 3.2; this collection occurred one month after the collection of access control information (see below), and so some small changes had occurred in group membership.

Limited information about the temporal evolution of group membership was obtained by looking at "created" and "last modified" times on group definitions, when available, and by collecting additional snapshots of DocuShare group membership data over a two month period about a year after the original data collection.

When possible, users and groups were characterized as *active*, *i.e.,* in current use, or *old* – this particular organization had a policy of maintaining all old user accounts in perpetuity (though with login disabled) to maintain access to necessary files. The sole exception to this was DocuShare, which limited the number of available user accounts, where old accounts were culled. Old accounts and/or groups, where identifiable were removed from some analyses. Similarly, as we were interested in sharing among groups of *humans*, counts of users and groups were filtered when possible to remove programs and machines, or groups designed only to contain programs or machines.

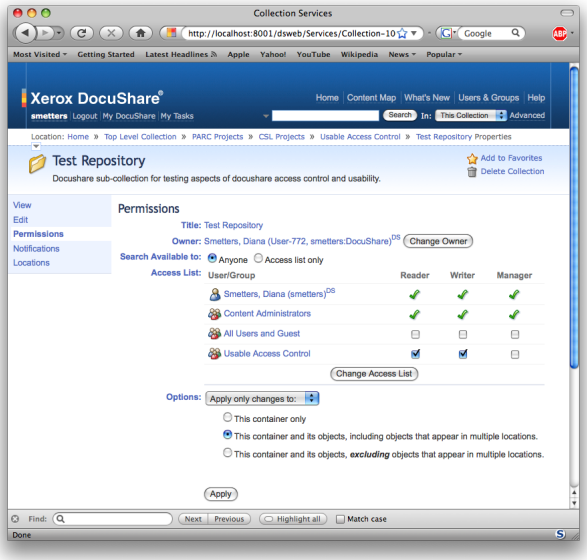

Figure 1: The permissions management interface in DocuShare. DocuShare, like most CMSs, can be accessed via a web browser or Windows file interface.

3.2 DocuShare Access Controls

To characterize the use of access control lists in a real document corpus, we retrieved anonymized document and access control information from our subject organization's (single) DocuShare server. DocuShare adopts a relatively simple access control model, designed with a strong focus on usability. Each piece of content in DocuShare is associated with a set of ACL entries, each granting access to a user or named group in terms of a small number of available rights, or permissions. These are read, write, manage (change permissions and ownership), and search. Unlike Microsoft Windows, it is not possible to explicitly *deny* someone a right (*e.g.,* to say "all members of group 'bowlers' *except* Bob can read this file"). This makes evaluation of access policies in DocuShare relatively simple – if a right is listed, it is granted, if not, it is not. In systems with explicit rights denial, effective access rights depend not only on each ACL entry alone but on the interaction between them, and in particular the order in which they are applied; this can be very difficult for users to understand [10, 13, 3]. Each object in DocuShare has a single *owner*, by default its creator. Owners have full, irrevocable access rights to the objects they own; ownership can be transferred to another user, but groups cannot own files.

Specification of policy can be simplified through the use of *policy inheritance* – an access control policy attached to a given folder (termed by DocuShare a "collection") automatically applies to documents added to that folder. Hence a user can create a particular

collection for the purpose of sharing files with a given group, and anything added to that collection is automatically made available to that group. When rights on a folder are changed, the user is prompted about how she wishes that change to be inherited.

In DocuShare, access rights do have one notable impact on the user experience – documents and folders which the user is not allowed to read are rendered effectively *invisible* to that user. While this seems to offer comfort to some users on an intuitive level ("if they can't see my files they must really be safe"), it amplifies the negative effects of permissions errors. As users lacking access to a document can't see it, they often don't realize that they are in fact missing something – sometimes until, say, others in a meeting start asking why they didn't read a key report.

Data Collection Our survey collected a highly anonymized snapshot of available documents and their access control settings on the organization's DocuShare server. Collection was done at a single point in time, avoiding potentially sensitive issues arising from looking at logs of users' activity.

In order to obtain informed consent and protect sensitive file information from even the researchers performing the study, we recruited eight DocuShare users as subjects in the study. Those subjects ran our data collection software using their DocuShare access credentials, giving us a "view" of all those documents to which they had access. Public documents will be seen by all subjects, while high privilege users will be expected to see a superset of the documents visible to others. The subject pool w selected to maximize the amount of data sampled, and consisted of regular users drawn from multiple internal sub-organizations, and several levels of managers with increasing levels of authority. Table 4 shows the number of documents visible to varying numbers of our subjects. Only a small number of files were visible to only one of our subjects. This suggests that in fact, we did collect data on the overwhelming majority of documents on this server, and adding additional subjects would not have revealed additional data.

We used a standard DocuShare Java API to communicate with the server, crawling the DocuShare content tree and enumerating every document (but not some specialized DocuShare objects such as calendar entries) to which the subject had access. This includes both private or protected objects to which the user has been granted access, and public objects accessible to every DocuShare user (and arbitrary "guests").

For each document or folder, the software recorded an anonymized representation of that object and all of the access control list (ACL) entries applied to it. Each piece of identifying information collected about an object – the name of a document, group or user, the unique DocuShare system identifier associated with each object, *etc.* – were replaced by an *anonymized content identifier*, or ACI. An ACI is computed as the 160-bit keyed cryptographic digest (SHA1-HMAC, [7]) of its argument; it is a unique representation of a given piece of data, but it is not possible to go "backwards" to recover that data from the ACI. Computing an ACI value for a given piece of data (*e.g.,* to check for a match) requires access to the appropriate anonymization (HMAC) key. The key used here was generated randomly at the beginning of the study and known only to the researchers; it was used throughout the study. The result were identifiers which were completely private, but which could be compared across subjects – *i.e.,* one can tell if two subjects have access to the same file, or belong to the same group, one just cannot tell what the name or contents of that file or group actually are. Only entries representing administrative users and groups, and built in groups representing public access were not anonymized; these groups have special properties and need to be separately handled in analysis. Anonymization was performed automatically before the data was submitted to the researchers, and subjects were given the opportunity to review their data at the conclusion of collection and decide whether to include it in the study.

3.3 Data Limitations

While our data set is notable for reflecting the behavior of a stable population of users over a very long period of time, it suffers from the fact that it comes from a single, relatively small organization, whose behavior may or may not be generally representative. Our subject organization is likely atypical in a number of respects: first, it has a high percentage of technically sophisticated users, although does include many users with limited technical skills. Second, it has perhaps lower turnover than a typical corporation, and as a result, may face a lower risk of insider attack. Finally, as a small, relatively flat organization, it may show less concern with internal divisions and security than a typical corporation. While it is important to keep these limitations in mind when evaluating our results, we should note that many of the existing studies in the literature looked at organizations with very similar properties to this one, which may in fact aid in comparison of results.

4. RESULTS

Users can assign access controls to a document directly or indirectly. The direct mechanism is to specify what *principals* – users or groups – have rights to that document, and what those rights are. The second, indirect mechanism, is to determine the makeup of those principals – which individuals have user accounts that can be referred to, and what groups exist and who belongs to them. We examine both in this study, beginning with an examination of groups.

4.1 Groups

Named access control groups serve several functions in document sharing. They can serve to separate specification of access policy on particular documents as reflected in individual ACLs, from the definition of organizational structure or roles as reflected in group memberships. Often different sets of users – document authors in the first case, managers or administrators in the second – are responsible for managing these two types of policy, and the level of indirection afforded by such groups allows them to do so autonomously. They also provide convenience, identifying sets of users shared with often enough to make it seem worth the cost of defining a group to represent them. As such, they represent a proxy for users' conscious sharing patterns.

To understand the role named groups play in sharing and access control, we performed a comparative analysis of the groups created for 4 separate sharing systems used by the same user community. These groups, described in Table 1 can be characterized by 1) whether users can define groups for themselves, or whether administrators must do so on their behalf, 2) whether groups can in turn contain other groups as members, and 3) whether membership is defined using a tool that can automatically validate the correctness of a request (*e.g.,* that an added user actually exists), or by editing a non-validating text file.

The four group types compared were DocuShare group memberships, Unix group membership for a large, NIS-based Unix community, Microsoft Windows domain group membership, and email mailing lists. Windows and Unix groups serve as the primary access control structures for two important content sharing mechanisms in this organization – networked file systems, one based on NFS and focused on Unix users, and one based on Microsoft's SMB protocol and used predominantly from Windows. Both of these group types were managed by systems administrators; if an

System	Defined By	Contain Groups	Specified By
DocuShare	users	yes	validating tool
Windows file sharing	administrators	no	validating tool
Unix/NFS file sharing	administrators	no	text file
Mailing lists	users	yes	validating tool

Table 1: Properties of different types of sharing groups for the organization under study.

end user wanted a new group to be created or an existing group to be altered they had to request an administrator make the change.

DocuShare provides both Web- and Windows file browser-based interfaces to document sharing, and is designed for user-friendly content management. It provides an access control model which is fully *discretionary* – end users can create and manage access control groups without requiring intervention by an administrator. It may therefore give a clearer view of what users want out of an access control system than more traditional, administrator-mediated approaches.

It may seem somewhat unusual to compare these traditional access control mechanisms to the use of mailing lists. However, given the overwhelming tendency of users to share documents by email [16, 17], email mailing lists (and the dynamic groups generated in the To and CC lists of each message) are in fact the most commonly used access control lists encountered by users. Mailing lists in this organization were also fully discretionary, managed by end users via a Web interface.

Both DocuShare groups and mailing lists (but not Windows or Unix groups) can contain other groups as members, potentially simplifying group definition and management. Both DocuShare groups and mailing lists also have a notion of an "owner", by default the user who created the group. DocuShare groups have only a single owner, while mailing lists can have many – corresponding to DocuShare's notion of "management" privileges. Each group in these discretionary systems 'specifies whether its membership can be changed by any group member (which we refer to as "open"), or only by the group owner ("closed"); administrators can also change group membership (and frequently do, for example to remove privileges from former employees). We show separately the results for all mailing lists and the subset of mailing lists which were closed, which will allow us to see any effect of the additional burden of placing access requests via an owner.

For each group type, we collected information about which users and groups comprised its members. When the information was available, we also recorded who owned each group, when it was created, when it was last modified, and by whom, and who could change its membership. Groups which contained other groups had their membership "flattened" for analyses of group size and the number of groups per user (*i.e.,* a member of a child group was treated as a direct member of a "flattened" parent group). When such information was available, inactive (as marked by administrators) user accounts, and user accounts and groups which represented machines rather than people were removed from analysis. The number of users/group omitted empty groups, which were usually no longer in use. Because these sanitization methods were inexact due to incomplete information, we provide only heuristic comparisons between populations here, rather than potentially misleading statistical tests.

4.1.1 Group Number and Membership

Table 2 shows the average number of members and memberships for groups and users in the various sharing systems under study.

While the average number of groups per user for the three file sharing systems is small, the distributions are highly skewed, and can be compared more effectively in Figure 2.

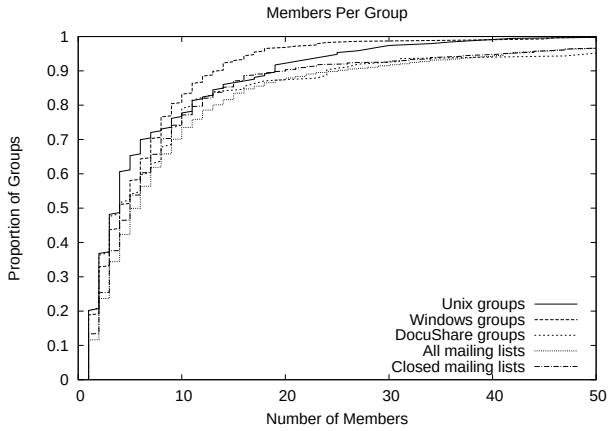

Figure 2: The distribution of group sizes for different types of groups.

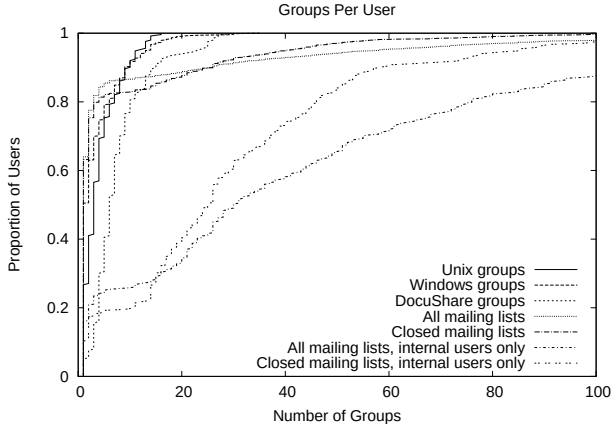

Figure 3: The distribution of the number of groups per user, for different types of groups.

Figure 2 shows the distribution of group sizes for different types of groups. There is a slight tendency for DocuShare groups and mailing lists to contain more members than Windows and Unix groups. This distinction may be partially artificial, as early versions of NFS prevents users from participating in more than 16 groups [1]. It may also represent the fact that DocuShare groups and mailing lists allow groups to contain other groups, making it very easy to build up large group sizes with less management overhead.

Figure 3 shows the distribution of groups per user, for different

[1] K. Farrar, personal communication.

System	# of Groups	Avg. Users/Group	Avg. Groups/User	Max G/U
Windows file sharing	324	6.3	3.7	35
Unix/NFS file sharing	193	7.0	4.2	16
DocuShare	131	11.9	7.8	30
All Mailing lists	1379	14.2	9.3	411
Closed Mailing lists	765	15.8	7.6	280

Table 2: The number of non-empty groups, and average number of group members (users/group) and user memberships (groups/user) for the various sharing systems.

types of groups. We can see that users participate in a large number of sharing groups. In one study [16], users self-reported that they shared data with 7 groups and individuals. The data presented here suggest that the number may be much, much larger, even just of "static" groups users are willing to generate explicit system representations for. While the four sharing systems described above are treated separately in our analysis, the population of people using them to share content is one and the same. It may be that the group definitions for each sharing system overlap completely – *i.e.,* there is a corresponding Windows group, Unix group, and mailing list for each DocuShare group, leaving the number of functional groups each user participates in as the intersection of the different group types. If they do not, however, the effective number may be closer to their sum.

DocuShare users belong to more groups than Windows and Unix users, as seen in the clear gap between the DocuShare curve and those for Windows and Unix. The most striking effect, however, is the sheer number of mailing lists users participate in. When all email users are considered, (two leftmost mailing list curves in Figure 3) there are two phases to the distribution of number of groups per user. There is a very large number of (usually external) users who participate in only a single large public mailing list, as seen in the steep initial portion of the curve. The remainder of the curve shows a long flat climb, representing many users who participate in a very large number of groups.

This behavior is clear looking at the distribution of groups per user for only members of the organization, excluding the large number of external users participating in a single mailing list. It provides a more representative sample of subjects for whom the majority of their work-related mailing list memberships can be seen. The resulting curves again show an initial sharp climb representing users participating in a single mailing list; these are typically users no longer with the organization whose group memberships have not been properly cleaned up. The remainder of the curves show that these users participate in extremely large numbers of mailing lists (47.5 or 31.3 closed groups, on average).

The net result is that users participate on average in more groups when they can define them themselves. They are clearly creating these groups for the purposes of limiting access to data. Fully 89.4% of DocuShare groups and 55% of mailing lists were *closed* – users wishing to participate in those groups had to be added to them by group owners or administrators. Taken together, this suggests that users not only need to share content with very large numbers of groups, but that they want to do so with some degree of control over where that content goes.

4.1.2 Group Structure

The ability to define a group in terms of other, member groups can make group management easier, at the cost of transparency – making it more difficult to determine who belongs to a group. Only two of our access systems, DocuShare and mailing lists, permitted groups as group members. Table 3 shows that this feature was only

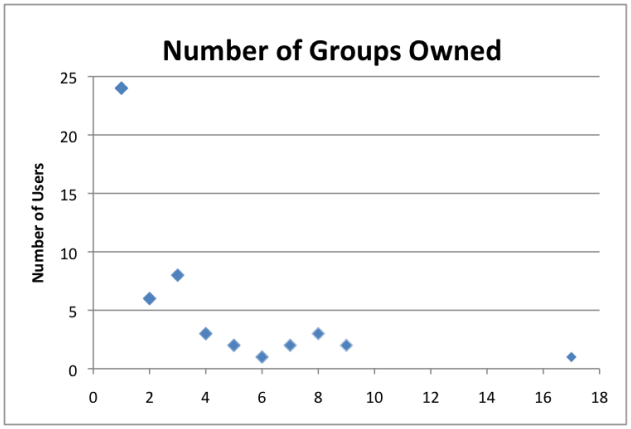

Figure 4: The number of groups owned by individual users.

used in about 20% of group definitions, and the overwhelming majority of group definitions included explicit user members. The complexity of group definitions was relatively limited, refering to a maximum of 4 (DocuShare) or 6 (mailing lists) levels of nested groups.

The ability to nest groups also resulted in likely access control specification errors. Some of these errors could result in data leaks. For example, there were three DocuShare groups whose only member was the effective group representation of public – a group named "All Users And Guest", representing anyone with network access to the web interface of the server. At the same time, the containing groups were closed, *i.e.,* limited to modification by their owners only, suggesting they were intended to be private; and only one of the three had a name indicating it might be intended to be public.

Even in systems where groups could not be nested, groups were often clearly related. For example, project groups might be broken into a core team and a larger group including ancillary members. These paired groups might be represented as two individually maintained, overlapping user lists, or by (when possible) including the core group as a member of the larger group. Both patterns were seen, even in systems where group definitions could be nested. This suggests that the ability to "nest" groups is merely a convenience feature, and may not be worth its cost in transparency; in fact this functional ability can be easily simulated by tools that help users maintain the relationships between coupled groups even in the absence of explicit nesting (*e.g.,* by storing relationship metadata with the group).

4.1.3 Group Management

The task of managing groups in Docushare is handled by small subset of user. Only 52 unique users of our population of 388 (13.4%) own groups. At least 39 users (10%) have modified groups.

	DocuShare	Mailing Lists
Total Number of Groups (incl empty)	159	1494
Groups With Groups As Members	29 (18.2%)	338 (22.6%)
Groups With Only Groups As Members	17 (10.6%)	40 (2.6%)
Maximum Nesting Depth	4	6

Table 3: Users made limited use of the ability to define groups in terms of other groups. As we study users' choices in group definition here, we include all user-defined groups (including empty ones) in our analysis.

Most group modifiers are also owners – only 56 users (14.4%) do either task. This may be an underestimate; our data shows only the current owner and most recent modifier of a group. The task of group management is shared, however, among this population – 32 users (57.1%) own groups which have been modified by others. Group ownership is fairly evenly distributed among this population; Figure 4 shows the number of groups owned by individual users. An administrative user owns 17 groups, presumably adopting them as their original owners have departed. The distribution of users modifying groups is similar. Administrative users are often the last people to modify the majorityof groups (82), suggesting that it is they, not the regular users, that take on the more mundane task of "cleaning up" old access rights.

This clean up (deleting or deactivating old user accounts, removing unwanted privileges,etc) is a common problem in securing sharing systems, and appears to be rarely performed unless there is some reason. There are many examples of "old" users and groups that still have access and influence long after the reasons for them are gone. We can see a number of classes of such "ghosts in the machine" in our sample; groups with no members, old user accounts that still belong to active groups, or groups corresponding to management functions for parts of the organization that no longer exist. In an extreme case, a user account belonging to a dead person was still active on the system. Interestingly, economic motivations may generate more effective clearing-up than security motivations – in our sample, old DocuShare user entries were regularly "cleaned up", probably due to the limited number of DocuShare account licenses available.

4.1.4 Group Dynamics

As our data was largely comprised of static snapshots, we have limited information about the way group membership changes over time. However, for DocuShare groups, we have information about group creation and modification times that let us estimate their dynamicity. We collected additional static snapshots of DocuShare group membership beginning one year after our original data collection, and continuing intermittently over a two-month window. We use this data to estimate the lifespan and dynamicity of DocuShare groups, and to get some hint about patterns in change of group membership.

The oldest group in our sample was 11.13 years old, the youngest, 4 months. Users actively modify groups. Only 12 groups (7.4%) never changed their membership at all; 19 groups (11.8%) were never modified after the day they were created (all 19 were over 6 months old at the time of data collection). Many groups have long active lifespans, as shown in Figure 5 – *e.g.,* for 7 groups, 9 years elapsed between group creation and most recent modification.Two groups showed membership changes within the last month, while one had gone 10.5 years since its last change. Only 19 groups changed membership within the last year.

The vast majority, 144 groups (89.4%) were "closed" – changeable only by their owner. Only 17 groups (10.5%) were changeable by their members. 14 groups (8.6%) were vestigial, containing no remaining members.

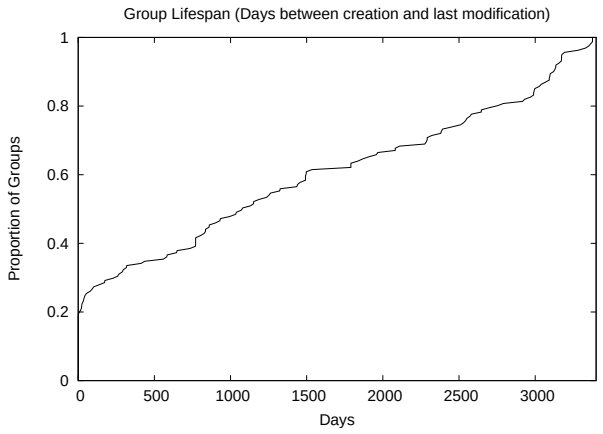

Figure 5: The distribution of the active lifespan of groups. This is an underestimate; measuring only up to the last modification at the time of data collection.

We looked at changes in DocuShare group membership over a 2 month window. In that period, there were only 6 group membership changes, made to 6 separate groups. A departing employee was removed from 4 groups, and 2 users were added to new groups. Interestingly, group managers performed both membership additions, while removals were largely (3/4) performed by IT administrators. This suggests a potential difference in users' approach to adding access – a gain of function, and removing it – often a "clean up" procedure after someone leaves a position or role. The latter process may be more often left to administrators for two reasons: first, users may be less concerned with security and so may not do what is largely a maintenance task, and second, it is easier for administrators to determine that departing employees should be removed from groups than to identify what current users need what specific additional access.

4.1.5 Group Meaning and Relationships

Finally, we examined the intended function of groups, as expressed in their names, as well as the relationship between them. Relationships between groups were indicated both by having one group be a member of another,or by creating groups with related names. In fact, group relationships were more often represented in names than in nesting, even when nesting was possible.

Use of an meaningfully-named group serves to indicate the user's *intent* in specifying an access policy. This can help others who wish to maintain the policy later remain true to that intent. For example, the same users might make up two separate groups: "Project Managers" and "Bridge Club". Granting access to those users via one of those named groups helps indicate in what role those users are to be granted access. Traditional role-based access control expects ad-

ministrators to specify roles in a top-down fashion, matching organizational structure and goals. What we saw in these user-defined groups was a similar functional categorization of individuals, but one that was generated bottom-up, according primarily to the needs of the users managing content, and not necessarily matching any predefined structure – effectively a *folksonomy*.

Like most folksonomies, they tend to be disorganized, duplicative, and decay over time, in this case with potential security consequences. This can be seen by comparing the administrator-managed Windows groups in our sample with the user-generated DocuShare and email groups. Windows groups had very clear structure and function. Access needs were translated directly into group structure to limit errors – for example, if there was a project titled "Foo", there might be two Foo-related groups. One called "Foo-Read" would contain members who should have only read access to Foo-related documents, and one called "Foo-full" whose members should have full control (write and manage access) to those same documents, making it clear how those groups should be used in acces control lists.

In contrast, our sample of DocuShare groups were much less organized. There were definitions for multiple groups representing the same likely user intent, but with different sets of members; for example a pair of groups with the same name but different capitalization. There were groups whose names were misleading; for example a group whose name indicated that it contained the organization's senior management contained only one member, a former senior manager. The current group representing the organization's senior management in fact had a very cryptic name likely only recognizable by its members themselves. This creates a very real risk of both errors in management, where users are added to or removed from incorrect groups, and errors in use, where the wrong groups or users are given access rights to documents.

4.1.6 Other Errors in Group Construction

Automated tools to aid users in group construction can be very valuable in preventing errors. In our analysis of Unix groups, which are created through editing a text file, we discovered a number instances of access control errors – accounts named incorrectly in a group membership list, usually through misspelling of an existing account (several of those instances actually concerned a single user). The result of such errors is that users do not end up with the access intended; even though both the access grantor and the administrator effecting the grant think access has been granted. Some instances of failed access grants concerned membership in "emergency" groups – groups designed to allow designated sets of users "break the glass" and act as systems administrators when necessary [**?**]; the discovery of such access errors would then be expected to occur at a critical moment when such access was most needed.

4.2 Access Control Lists

We want to understand when users will actually go to the effort to explicitly manage access, and how much effort they will go to. In section 4.2.2 we look at how many documents and folders in DocuShare had their permissions explicitly set – changed from what they would have automatically inherited from their parent (containing folder). In section 4.2.3 we look at the types of access control lists users actually construct for their content, and attempt to discover any commonly used "patterns" of permissions. To begin, we provide a brief overview of the set of documents in our sample.

4.2.1 Documents

Our automated analysis retrieved data about 49,672 unique objects – documents or folders visible to some or all of our eight subjects. Of those, 42,569 were accessible to all subjects, and 3,375 were accessible to only one. Table 4 shows the number of documents visible to each of our subjects, and the number of subjects who could "see" each document. The limited number of documents made visible by adding additional subjects suggests that our data pool does contain the majority of documents present on the server.

Subject	# of Documents	# of Users	# of Documents
1	45,859	1	3375
2	45,329	2	762
3	43,500	3	250
4	45,670	4	57
5	42,922	5	1742
6	47,069	6	40
7	44,156	7	877
8	47,150	8	42,569
Total	49,672		

Table 4: The number and types of documents (files and folders) visible to each of the eight subjects in our study, and the number of documents visible to a particular number of users.

Type	Count	Percentage
File	36139	72.7%
Folder	7141	14.3%
URL	3365	6.7%
Event	2854	5.7%
Calendar	101	0.2%
Bulletin board	72	0.1%
Total	49,672	

Table 5: The number and types of documents studied.

Table 5 shows the distribution of object types in our data set. The vast majority (72.7%) were simple files, while 14.3% were folders (termed *Collections* in DocuShare). The rest were specialized object types used to store URLs, calendar entries, *etc.* . In the remainder of our analysis we often compare the properties of folders with those of all other document types (files, URLs, events, *etc.*) grouped together and referred to as "documents". Figure 6 shows the distribution of folder sizes in our data set.

The oldest object in our sample was a folder created in November of 1996, which was last modified 9 years later, in May of 2005 (our data collection was performed in late December 2007/January 2008). The oldest folder still in active use was created in September of 1997, and was modified in December of 2007.

Updates to documents in DocuShare create new "versions" – providing new content without changing the name of the document or the external links to it, and leaving the old versions still accessible. Only 10,107 files (20.3%) had more than one version – the majority of files are not changed after they are uploaded. A few documents have very high version counts – for example, one document,probably a monthly report, went through 71 versions over the course of a single year. The first was created by the original document owner, the remainder by a new owner who took over the document, and uploaded new versions approximately every month.

DocuShare also offers the ability to represent an object in more than one location – *i.e.,* to have more than one parent in the file hierarchy, without producing a second copy that must be separately maintained. This is not a heavily used feature; only 391 documents live in 2 locations, 38 in 3 locations, 8 in 4 locations and 3 docu-

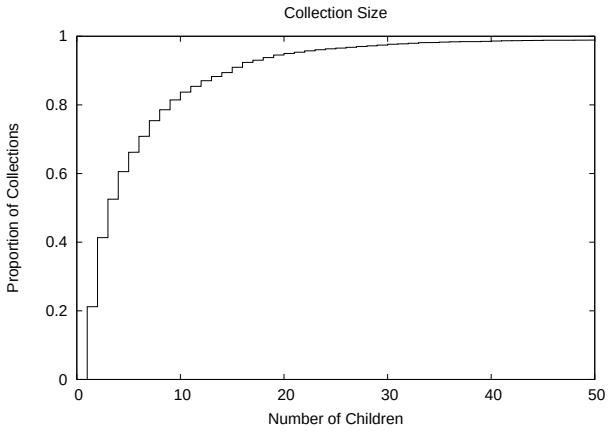

Figure 6: The distribution of the sizes (number of children) of folders, for the 7141 in our sample. 74 folders (1%) had more than 50 children, with a maximum of 804; 1446 folders had only 1 child.

ments in 5 locations. For one subject, anonymized content identifiers (ACIs, see section 3) were computed over the content of the documents available to that user; revealing 1875 documents stored on the DocuShare server more than once, either as repeated versions of the same document with the same content, or as separate named documents.

4.2.2 Setting Access Rights

We want to understand how often users actually go through the trouble to set access permissions on documents and collections. By default, a document or folder in DocuShare will inherit the permissions of its parent. We can therefore find where users explicitly set permissions on content by looking for files and folders whose access control lists differ from those of their parents. We ignore those differences in ownership which arise automatically as multiple users add content to a folder, each by default the owner of the content they had added.

Out of our sample of 49,672 documents and folders, we found only 2,608 cases (5.2% of all objects) where access controls were explicitly set or modified – documents or folders whose access control lists differed from those of their containing parent folder. The remaining 95% of objects simply inherited the permissions associated with the folder in which they were stored. Of these modified objects, 1076 were folders and 1531 were documents. Given the total numbers of documents and folders given in Table 5, only 3.5% of documents (1531/42531) had their access control lists modified, while at the same time users explicitly set access controls on a relatively large proportion, 15% of folders (1076/7141).

When access control lists did differ, they could do so in one of two ways – first, the list of principals (users or groups) given explicit access could change, as additional principals were added to or principals were removed from the child's access control list. Second, for a given principal that existed on both lists, the permissions, or access rights associated with that principal could change – *e.g.,* a child could grant write access to a principal to whom its parent only granted read access. Of our 2,608 cases of changed access rights, there were 1,366 cases (52.3%) where the set of principals named differed between the access control list of parent and child, and 791 cases (30.3%) where the permissions associated with a given principal were different. In 451 (17.2%) cases, both principals and permissions were changed.

These results confirmed our expectations that users rarely change access rights on a folder or file, preferring to add new content to an existing folder with its permissions already set, and inherit the permissions of that folder. This reluctance to change permission settings could be due to the fact that most users simply do not need to – that the cases where permission changes are necessary are rare, and handled by setting the permissions once on a parent folder, using inheritance to take care of the rest. This would be consistent with "best practice" recommendations for using access control settings, which recommend setting them rarely and relying on inheritance to manage most controls [3]. Alternatively the reluctance to change permission settings could be due to limitations in the user interface (shown in Figure 1), making it simply difficult to do so. This is consistent with the results of laboratory studies [10, 13] which show that users make large numbers of errors when attempting to use traditional access-setting interfaces.

Though percentage-wise, users made the vast majority of their permissions settings on folders, we still wondered why users set permissions directly on such a large number (1531) of documents. We hypothesized that some of those apparent cases of active permissions setting could in fact represent *inheritance failure* – when a user changes an access control list on a folder in DocuShare, she is confronted with three choices for how that change should be inherited by the contents of that folder (and recursively, their contents). By default, the change applies to the folder, and all of its contents regardless of whether those contents appear in multiple locations (have multiple parents). The user may, however select one of two other alternatives – applying the change to the folder only, or applying the change to the folder and those of its children who appear in only a single location. Ignoring this latter alternative as so few documents appear in multiple locations (see Section 4.2.1), we looked for evidence that users may have sometimes applied access changes to folders only, resulting in folders whose access control lists differed from those of their children (folders and documents), and creating the appearance that those children had their access control settings manipulated directly. In such cases, all of the children of an altered folder should have identical access control lists (modulo differences in ownership if different users created those documents). Searching for folders with these properties, we found 326 of our 2608 objects who may not have had their permissions explicitly set, but instead been subject to such inheritance failure. 86 of these were collections, and 240 were documents. If all these were indeed inheritance failure, we would have only 2282 changed objects (4.5% of total), 994 changed folders (13.9% of total), and a slightly lower 1291 changed documents (3% of total). This metric tends to underestimate the amount of inheritance failure as it requires for detection confidence that changed nodes have more than one child.

4.2.3 Structure of Access Control Lists

Users only infrequently go to the trouble to explicitly set access rights. We wanted therefore to examine in greater detail *what* access control settings they did make, when they made them – how long were access control lists, and how did users make use of access control groups, as opposed to naming specific individuals and granting them access rights?

We analyzed the access control lists specified for documents and folders in DocuShare. For those objects whose access control lists were explicitly set, rather than inherited (see Section 4.2.2), there were 4,527 access control entries (not documents) that granted access rights to particular named users (apart from the built-in owner access), and 4,755 entries that granted access to groups (outside of the automatic access for the Content Administrators group). 2,090

Docs	Groups							
Users	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	8	282	478	231	23	2		
1	4	199	322	80	28	5		
2	5	284	177	30	17			
3	1	67	71	4	1	5		
4	2	37	12		2			
5	1	18	17	1	1			
6	1	8	3					
7		3	12					
8	2	19	11	7			1	14
9		3	1	1				
10		11	2					
11		9						
12		55	1					
13		8						
14		2						
17				1				
21			3					
26			1					
28			1					
29			3					

Docs	Groups							
Users	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	8	1939	6032	2741	451	43		
1	115	4200	5955	2454	186	328		
2	1074	5110	3242	582	136	43		
3	847	559	959	195	1	25		
4	3	2636	118		16			
5	1	2127	248	6	1			
6	15	70	133					
7		52	94					
8	6	180	356	41			1	17
9		20	476	9				
10		341	341					
11		104	6					
12		4867	1					
13		121						
14		12						
17				13				
21			3					
26			4					
28			1					
29			7					

Table 6: ACL Complexity. This table shows the number of user (rows) and group (columns) ACL entries in ACL specifications. a) The left table shows the distribution of ACL entry types for objects whose ACLs were explicitly set (out of a total of 2,608, see Section 4.2.2). b) The right table shows the number of objects effected by ACLs of a given complexity – the total count of objects (folders and documents, permissions set or inherited, out of a total of 49,672) with ACLs containing a given number of user or group entries. Default entries for owner and the Content Administrators group are omitted. Note that user counts skip values with no non-zero counts to save space.

of these entries (22.5%) allowed some principal read-only access to an object, while 6,766 (72.8%) allowed read-write access. Finally, there were no specifications allowing write-only (no read) access.

Our initial hypothesis was that most access control lists would be short and simple, to minimize user effort. Going further, given the intuition that people may share content with a single group of recipients at a time, we expected that the most common access control list structure would contain a single non-default entry, granting access to a particular named group. We expected that grants of access to particular individuals would be uncommon, as they would require greater user effort to specify, and incur follow-on management costs [3]. The finding that users do actively create and manage DocuShare groups (see section 4.1) would seem to support this hypothesis, as it would allow users to take advantage of work they and others had done in creating named groups.

We were surprised to discover that complex access control lists were the norm, rather than the exception. Table 6 shows the count of documents and collections with a given number of principals (users and groups) in their access control list, for both objects with explicitly-set ACLs and the total object population which inherits their behavior; where ACL entry counts are given separately for entries granting access to groups (rows) and users (columns). Default entries for object owners and the Content Administrators group are omitted. In particular, there were only 1,939 documents (3.9% of sample) that had ACL specifications of the form we expected, with only a single (non-owner, non-administrative) entry for a named group.

4.2.4 Access Control Patterns

Table 6 shows that ACL specifications in DocuShare are often more complex than predicted. We can then ask the question of how they got that way – do users truly find such complex access policies necessary, or are there hidden patterns here which make complex-seeming ACL specifications in fact much more simple?

Errors. One of the simplest and most common reasons for generating complex ACL specifications is simply human error. Many of the access control lists in our sample contained repeated information – *e.g.,* explicit mention of users or groups who were already granted access. For example, in all cases with 17 or more individuals named on an ACL, each of those objects also contained an ACL entry granting access to the large "authorized" users group, of which all the explicitly named individuals were already members. These repeated specifications of a given principal in an ACL or group definition may seem merely inefficient, but in fact they can lead to more severe errors – a user attempting to revoke a given principal's access may remove only a single entry for that multiply specified principal from a group hierarchy, leaving others; or may remove the principal from a known group, missing a file on which they are individually granted access. It is also not clear whether the effective policies users set on their documents were what they intended, or instead what they arrived at in an attempt to reach their functional goals. For example, users may *over-grant* access – adding rights for additional users and groups either to repair access failures or to preemptively avoid them – until they arrive at some combination that allows all desired access. The fact that such rules also allow a great deal of undesired access [6] is unlikely to be noticed given current interfaces.

Access Design Patterns. Another reason for complex ACL specifications could be that they are, from the user's point of view, actually simple – they represent policies which when expressed in natural language would seem much simpler than their implementation. We could consider these *access design patterns*.

For example, the simplest access control model that can be ap-

plied to a document is to make it *private* – accessible only to its owner. From Table 6 we can see that our analysis found only 8 such documents – documents with 0 user ACL entries and 0 group ACL entries – which would be accessible only to their owners and the Content Administrators group. However, this pattern appears, for this DocuShare server, to be vanishingly rare – not only do only 8 documents match that access pattern, 5 of those are empty folders, leaving only a single folder with 2 children to fit this picture. Another design pattern seen most commonly in our Windows group data is to define separate groups for different access control functions – *e.g.,* individuals who should have read access to project documents, vs individuals who should have read-write access. This pattern would result in ACLs containing multiple group entries, one for each group playing a functional sub-roles.

Public Access. Another common access control pattern is to make data *public*. All content sharing systems have some concept of sharing with "everyone", but differ in the way that "everyone" is expressed. For example, DocuShare provides two built in "everyone" groups – "All Users and Guest", which refers to anyone accessing the DocuShare server, whether they have logged in or not; and "All Users Except Guest", which refers to any registered user of DocuShare. Additionally, our subject organization two almost identical groups representing "authorized" users, and containing effectively every DocuShare user in the organization.

Of the documents we studied, 11,302 (22.7%) were readable by one of the two built-in "everyone" groups. Interestingly, 8,755 (77.4%) of these were also publicly *writable*, including 178 which were writable by "All Users and Guest"; these were meant to be truly public. If we expand consideration to the two local "public" groups, a very large percentage of the documents in our sample are effectively public – 42,748 documents (86%) are readable by at least one of these 4 groups, and 33,397 (67.2%) are writable by at least one of them. The high proportion of publicly accessible content on this DocuShare server may reflect one of two things: first, that this organization, which has a relatively open culture, errs on the side of publicly sharing data, or alternatively, that DocuShare is predominantly used for web-based distribution of largely "public" data; with more limited-access data being shared by other means.

The large number of publicly accessible documents reinforces the notion that the ACL specifications in table 6 are unnecessarily complex. Other than allowing more closely held control over write access, a publicly-readable document does not need to specify long, superfluous lists of additional users and groups allowed to read it.

5. CONCLUSIONS

In this study we find, perhaps unsurprisingly, that users almost never set access control policies on content, preferring instead to rely on *context* – inheritance of policy from the location in which the content is stored. They share that content, though, in complex ways. Users participate in much larger numbers of access control in email sharing groups than self-report studies have suggested they do, particularly when they can define those groups themselves. We hypothesize that this may be because as groups get easier to make, users will want to make more and more groups to handle their complex information demands, as in the case of mailing lists.

We also find that the access control policies that users apply to their content (either by direct definition, or via inheritance) are quite complex, as measured by the number of individually specified grants of access rights. More careful analysis of our data suggests that in part this is due to user error – users end up defining policies with effects other than what they might have intended, or redundant policies that could in fact be expressed in much simpler ways. This is consistent with the results of laboratory studies which suggest that users make many errors when defining access control policies using traditional interfaces [10, 13]. We hypothesize that the remainder of this complexity might be reduced by considering these policies as examples of simpler *access control patterns*.

While these results come from a single, small organization, we suspect that many of them are quite general. In future work we hope to extend our analysis to other organizations for comparison.

5.1 Implications For Design

By examining how the users in our study make use of the access control tools available to them, we can derive a number of suggestions for the design of both access control systems themselves, and the interfaces used to manage them. We summarize these here. In future work we hope to evaluate their effectiveness in real systems.

5.1.1 Simplify Access Control Models

We would argue that to be user-friendly, access control systems need to limit the flexibility they offer in order to simplify use. Based on our data, we suggest the following limits on access control models:

Only allow positive grants of access. Adding explicit access denial takes a system where the impact of an access rule can be considered in isolation to one where access depends on the combination of rules in effect and the order in which they are applied. [3, 13]. This adds tremendous complexity, significantly limits the effectiveness of clever user interface techniques to simplify access management [14, 9], and offers no significant functionality that cannot be otherwise achieved, say via more complex group definitions. Many systems get along fine without access denial (*e.g.,* DocuShare, email, Unix permissions), and those systems that do include them discourage their use [3].

Simplify the inheritance model for access control changes. When a user changes access rights on a folder, they are typically confronted with a choice of how those changes should propagate to the folder's children. That model of access inheritance directly mirrors the implementation choices faced by developers, but does not mirror users' mental models of how inheritance should work. Users seem to treat the hierarchy of content covered by a folder's access settings as a common protection domain, operating together until overridden by any explicitly-set ACLs present below. A simple default inheritance scheme might apply ACL changes to all content in the current protection domain, supplemented by visualizations to help users explore the extent of their changes.

Limit the types of permissions that can be granted. There seems an ever-constant expansion of the types of access rights that can be managed. While separation of read and write and perhaps execute permissions are clearly valuable to users, it is not clear that others (*e.g.,* separate control of access settings themselves, deletion, or other options) are.

Group Definitions The one facet of access control management users seem relatively comfortable with is the creation and management of groups, and in fact creation of appropriate group structures can emulate many other, more complicated means of achieving an effective access control policy. Allowing users to flexibly create groups to meet their needs seems the simplest route to providing needed flexibility, rather than attempting to specify a small number of predefined groups and allowing other, more complex access specifications.

5.1.2 Improve Tools For Managing Access

There are a number of areas where improved tools for managing access could significantly decrease error and increase user and

administrator effectiveness, including:

- Tools for group management that reduce redundancy and error in group definitions, and track the intended relationship between groups.
- Tools for ACL management that maximize the use of groups, and help generate concise ACL statements granting only necessary rights.
- Tools for administrators to manage access policy, directly focused on "cleaning up" outdated users, groups and permissions.
- Activity-based folksonomies of groups and users to help users choose the right principals with whom to share among potentially similar groups, and make old groups "fade away" naturally.
- Visualizations to enable users to see who has access to the content they are sharing, and what content is impacted by a change in policy.

We argue that ideally, these tools will operate in the context of the users' primary application task (*e.g.,* [9]), rather than as a separate interface focused only on policy management (*e.g.,* [10, 2, 13]). We believe that by simplifying the underlying the access control model as described above, we will have the additional side benefit of enabling more user-friendly interfaces.

6. ACKNOWLEDGEMENTS

The authors would like to thank Elizabeth Churchill, Les Nelson, Brinda Dalal, Victoria Bellotti, Keith Farrar and the DocuShare product team for useful discussion, and Elizabeth Churchill for suggesting the term "cyberethnography". We are grateful to Xerox for their support for portions of this work.

7. REFERENCES

[1] S. Ahern, D. Eckles, N. S. Good, S. King, M. Naaman, and R. Nair. Over-exposed?: privacy patterns and considerations in online and mobile photo sharing. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 357–366, New York, NY, USA, 2007. ACM.

[2] X. Cao and L. Iverson. Intentional access management: making access control usable for end-users. In *SOUPS '06: Proceedings of the second symposium on Usable privacy and security*, pages 20–31, New York, NY, USA, 2006. ACM.

[3] M. Corporation. Best practices for permissions and user rights, January 2005. `http://technet.microsoft.com/en-us/library/cc779601.aspx`.

[4] D. Ferraiolo and R. Kuhn. Role-based access controls. In *15th NIST-NCSC National Computer Security Conference*, pages 554–563, 1992.

[5] Flickr. `http://www.flickr.com`.

[6] N. S. Good and A. Krekelberg. Usability and privacy: a study of kazaa p2p file-sharing. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 137–144, New York, NY, USA, 2003. ACM Press.

[7] H. Krawczyk, M. Bellare, and R. Canetti. RFC 2104: HMAC: Keyed-hashing for message authentication, Feb. 1997. Status: INFORMATIONAL.

[8] S. T. K. Lam and E. Churchill. The social web: global village or private cliques? In *DUX '07: Proceedings of the 2007 conference on Designing for User eXperiences*, pages 1–7, New York, NY, USA, 2007. ACM.

[9] E. Lieberman and R. C. Miller. Facemail: showing faces of recipients to prevent misdirected email. In *SOUPS '07: Proceedings of the 3rd symposium on Usable privacy and security*, pages 122–131, New York, NY, USA, 2007. ACM.

[10] R. A. Maxion and R. W. Reeder. Improving user-interface dependability through mitigation of human error. *Int. J. Hum.-Comput. Stud.*, 63(1-2):25–50, 2005.

[11] A. D. Miller and W. K. Edwards. Give and take: a study of consumer photo-sharing culture and practice. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 347–356, New York, NY, USA, 2007. ACM.

[12] J. S. Olson, J. Grudin, and E. Horvitz. A study of preferences for sharing and privacy. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1985–1988, New York, NY, USA, 2005. ACM.

[13] R. W. Reeder, L. Bauer, L. F. Cranor, M. K. Reiter, K. Bacon, K. How, and H. Strong. Expandable grids for visualizing and authoring computer security policies. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1473–1482, New York, NY, USA, 2008. ACM.

[14] J. Rode, C. Johansson, P. DiGioia, R. S. Filho, K. Nies, D. H. Nguyen, J. Ren, P. Dourish, and D. Redmiles. Seeing further: extending visualization as a basis for usable security. In *SOUPS '06: Proceedings of the second symposium on Usable privacy and security*, pages 145–155, New York, NY, USA, 2006. ACM Press.

[15] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.

[16] S. Voida, W. K. Edwards, M. W. Newman, R. E. Grinter, and N. Ducheneaut. Share and share alike: exploring the user interface affordances of file sharing. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 221–230, New York, NY, USA, 2006. ACM.

[17] T. Whalen, D. Smetters, and E. F. Churchill. User experiences with sharing and access control. In *CHI '06: CHI '06 extended abstracts on Human factors in computing systems*, pages 1517–1522, New York, NY, USA, 2006. ACM.

[18] M. E. Zurko, R. Simon, and T. Sanfilippo. A user-centered, modular authorization service built on an RBAC foundation. In *IEEE Symposium on Security and Privacy*, pages 57–71, 1999.

[19] M. E. Zurko and R. T. Simon. User-centered security. In C. Meadows, editor, *New Security Paradigms Workshop*. ACM, 1996.