Carnegie Mellon University

CARNEGIE INSTITUTE OF TECHNOLOGY

THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF Doctor of Philosophy

TITLE **A Tag-Based, Logical Access- Control Framework for Personal Data-Sharing**

PRESENTED BY **Michelle Mazurek**

ACCEPTED BY THE DEPARTMENT OF

Electrical and Computer Engineering

Greg Ganger — ADVISOR, MAJOR PROFESSOR — 5/14/14 — DATE

Jelena Kovacevic — DEPARTMENT HEAD — 5/14/14 — DATE

APPROVED BY THE COLLEGE COUNCIL

Vijayakumar Bhagavatula — DEAN — 5/14/14 — DATE

A Tag-Based, Logical Access-Control Framework for Personal File Sharing

Submitted in partial fulfillment for the requirements for
the degreee of
Doctor of Philosophy
in
Electrical & Computer Engineering

Michelle L. Mazurek

B.S., Electrical Engineering, University of Maryland
M.S., Electrical & Computer Engineering, Carnegie Mellon University

Carnegie Mellon University
Pittsburgh, PA

May 2014

For my dad, Niel Mazurek, who made me believe that being an engineer is kind of like being a superhero. I miss you.
And for MG. I love you already and I can't wait to meet you.

Keywords: access control, file systems, usability

Abstract

People store and share ever-increasing numbers of digital documents, photos, and other files, both on personal devices and within online services. In this environment, proper access control is critical to help users obtain the benefits of sharing varied content with different groups of people while avoiding trouble at work, embarrassment, identity theft, and other problems related to unintended disclosure. Current approaches often fail, either because they insufficiently protect data or because they confuse users about policy specification. Historically, correctly managing access control has proven difficult, time-consuming, and error-prone, even for experts; to make matters worse, access control remains a secondary task most non-experts are unwilling to spend significant time on.

To solve this problem, access control for file-sharing tools and services should provide verifiable security, make policy configuration and management simple and understandable for users, reduce the risk of user error, and minimize the required user effort. This thesis presents three user studies that provide insight into people's access-control needs and preferences. Drawing on the results of these studies, I present Penumbra, a prototype distributed file system that combines semantic, tag-based policy specification with logic-based access control, flexibly supporting intuitive policies while providing high assurance of correctness. Penumbra is evaluated using a set of detailed, realistic case studies drawn from the presented user studies. Using microbenchmarks and traces generated from the case studies, Penumbra can enforce users' policies with overhead less than 5% for most system calls. Finally, I present lessons learned, which can inform the further development of usable access-control mechanisms both for sharing files and in the broader context of personal data.

Acknowledgments

As I do all things, I didn't do it alone. I did it with the help of a lot of people.—Sonia Sotomayor

This dissertation, and my academic career in general, could not have been accomplished without the help, support, friendship and love of some amazing people. I will always be grateful to the senior PDL grad students who generously provided their time, help, and advice when I was first starting out at CMU, including Mike Abd-El-Malek, James Hendricks, Brandon Salmon, and Shafeeq Sinnamohideen. Others who have generously listened to me practice talks, answered my questions, and been wonderful colleagues include Jim Cipar, Limin Jia, Elie Krevat, Yannis Mallios, Iulian Moraru, Wolf Richter, Divya Sharma, Yuan Tian, and Alexey Tumanov.

Eno Thereska gave me the opportunity to live in England for three months, spend time with the wonderful people at Microsoft Research Cambridge, and learn about research while working on a great project.

Nothing at all would ever get done without the help and support of the PDL and CyLab admin and IT staff, including Tonya Bordonaro, Kelley Conley, Bill Courtright, Joan Digney, Jenn Engleson, Toni Fox, Mitch Franzos, Zis Economou, Sami Stevick, and Tina Yankovich. Michael Stroucken saved my experiments (and my sanity) on a regular basis. Karen Lindenfelser, PDL den mother extraordinaire and friend, always made life easier and more fun. Nitin Gupta has been a wonderful cube-mate, collaborator, doer of annoying programming tasks I got to avoid, and most importantly, friend.

The passwords team changed my research life (only sometimes driving me crazy in the process); thanks to Nicolas Christin, Saranga Komanduri, Sean Segreti, Rich Shay, Tim Vidas, and the other team members for teaching me about research and about collaboration. I'm also glad I had the opportunity to work with and spend time with Iulia Ion, Christina Johns, Yuan Liang, Billy Melicher, Ilari Shafer, and Kami Vaniea.

There's nothing better than when great collaborators are also good friends. I have been lucky enough to share both work and life with Patrick Kelley, Peter Klemperer, Raja Sambasivan, Manya Sleeper, and Blase Ur. Other people who made life in Pittsburgh

more enjoyable include Anna and Matthew Beckler, Sam Gottlieb, Adam Hartman, Lauren Heller, Peter Milder, Caitlin Moyer, Bonnie and Matt Tesch, and all the members of the ECE GigaHurts softball team and their partners. Kristen Dorsey has been there for cooking experiments, jokes, qual cupcakes, support and advice during the tough times, and celebrations of the good times. For Kristen: "We're going! We come from a line of strong women!"

Thanks to the staff and students at the Pittsburgh Project for giving me something to look forward to every week between September and May, making me laugh, and reminding me why I love science. You haven't lived until you've helped a bunch of nine-year-olds blow up soda with Mentos.

I can't say enough to thank the members of my committee. Mike Reiter, Lorrie Cranor, and especially my advisers Greg Ganger and Lujo Bauer, have taught me about research, given me advice, provided feedback on countless drafts of my various papers, posters, talks, and application materials, given me pep talks when I needed them, provided me with amazing opportunities for networking and advancement, written me recommendation letters, and generally been everything a Ph.D. student could ask for. If I can be half the adviser to my students that you have been to me, I will consider it a great accomplishment.

My life is better every day because of the families I chose a long time ago. Iris Litwin, Rachel Tehrani, Dani Blum, and Shoshana Gleit—no matter how many months we go without seeing each other, we always pick up right where we left off. The Ellicott crew let us crash in their spare bedrooms and never stopped inviting us to parties and cheerleading for us to move back to Maryland eventually. (It worked!) I can always sing about poodles and celebrate shrubbery with Bruce and Lauren Webster, Amy Smith and Mike DePalatis, Jen and Mike Sheer, Keavin Stryker, Sarah Shaffer, Jon Roy and Liz Everson, Lauren and Larry Bardelli, Doug and Carla Wardell, and Jake and Tabytha Schwartz.

I am lucky enough to have some pretty terrific in-laws—thanks to Ruth, Marty, Paige, and Dan for your support and good humor throughout this process.

Everything I do is built on the unconditional love and support of the family I was born with. My grandmother Emmi Loewenstern is my role model and the toughest person I know. She taught me to always let the bad things go and treasure the good things. My sister Allison is always there to understand our particular brand of insanity, laugh with me, and provide long-distance shopping consultations. My mother Barbara showed me by example how commit completely to my goals, and taught me that people are the most important resource, a lesson which has never failed me. She has good advice for basically every situation. Mom, Allison, Grandmom, and Aunt Linda, thank you for calling to check on me; always sending me back to Pittsburgh loaded down with food; making me laugh; listening to my panics, rants, and triumphs; and most importantly, always believing in me.

Finally, to my wonderful husband and best friend, Kyle, who when I came home one Tuesday and said, "How about if instead of house shopping, I quit my job and go get a Ph.D?" said yes without hesitation. Thanks for making me laugh, bringing me dinner when I'm on deadline, doing the laundry, keeping me company, taking care of me through the worst times, and celebrating all the best times. You make the calm space in my life that allows me to take on new challenges. I'm so excited to start the next part of our adventure.

The research described in this thesis was made possible by the National Science Foundation via grants #CNS-0831407, #CNS-0756998, and #DGE-0903659; CyLab at Carnegie Mellon under grants DAAD19-02-1-0389 and W911NF-09-1-0273 from the Army Research Office; gifts from Intel, Microsoft Research, and Cisco Systems, Inc.; a Lamme/Westinghouse ECE Graduate Fellowship, a Facebook Graduate Fellowship, and a Parallel Data Lab Entrepreneurship Fellowship; and the members and companies of the Parallel Data Lab Consortium (including Actifio, American Power Conversion, EMC, Facebook, Fusion-io, Google, Hewlett-Packard Labs, Hitachi, Huawei, Intel, Microsoft Research, NEC Laboratories, NetApp, Oracle, Samsung, Seagate, Symantec, and Western Digital).

Contents

List of Figures

List of Tables

1 | Introduction

Modern technology is moving more and more of our lives into the digital world. People create and share a lot of personal files, conduct their personal and business activities online, and interact with increasingly ubiquitous sensors. Personal files are thus distributed across devices (laptops, tablets, phones, etc.) and the cloud (Gmail, Facebook, Flickr, etc.). Users are interested in accessing their content seamlessly from any device, as well as in sharing at least some parts of it with family members, friends, co-workers, and even the general public. As a result, systems and services designed to meet these needs are proliferating [13, 93, 106, 108, 113, 128].

Although these advances in storing and sharing personal files provide tremendous benefits to users, they also create risks to security and privacy via unexpected or unintended disclosure. News headlines repeatedly feature access-control failures with consequences ranging from embarrassing (e.g., students accessing explicit photos of their teacher on a classroom iPad [62]) to serious (e.g., withdrawal of college admission [63], or a fugitive's location being revealed by geolocation data attached to a photo [141]). As more data is created and made available via the internet, the potential for such problems will only grow.

Access-control failures generally have two sources: ad-hoc security mechanisms that lead to unforeseen behavior, and policy specification mechanisms that do not match users' mental models. Commercial data-sharing services sometimes fail to guard resources entirely [39]; often they manage access in ad-hoc ways that lead to problems [87]. Numerous studies report that users do not understand current products' privacy settings or cannot use them to create desired policies (e.g., [38, 66]). Popular websites abound with advice for these confused users [99, 119].

Simply refusing to share personal data is not feasible or desirable, but sharing indiscriminately is equally problematic. As the amount and complexity of available data proliferates, more management is required to properly control access with appropriately fine granularity. Yet, at the same time, access-control configuration is a secondary task, with nebulous future consequences, making it something most users do not want to spend much time on [45].

Better systems would allow users to efficiently accomplish their primary goals with-

out unnecessarily compromising their privacy. To make this possible, we must develop usable access-control mechanisms that allow users to decide when and with whom their information is shared, without requiring extensive time and effort for configuration and management. The gap between users' complex privacy and security needs and their limited time and ability to manage them is large. Bridging it will require insight and approaches from across the security, systems, and HCI communities: understanding users; needs, understanding how security properties relevant to these needs can be enforced, and building systems from the ground up to support both. In addition, we must collect real-world data that is often difficult to acquire; only by starting from high-quality data can we design systems and policies that will work for ordinary users.

Many attempts to reduce user confusion focus only on improving the user interface (e.g., [67, 110, 134]). While this is important, it is insufficient—a full solution also needs the underlying access-control infrastructure to provide principled security while aligning with users' understanding [46]. Prior work investigating access-control infrastructure typically either does not support the flexible policies appropriate for personal data (e.g., [51]) or lacks an efficient implementation integrated with system infrastructure (e.g., [83]).

In this thesis, I demonstrate a new approach toward the ideal of low-effort, high-accuracy access control for personal data, designed to support the policy preferences of non-expert users at the file-system-infrastructure level. The following statement summarizes the contributions of this thesis:

A distributed, file-system-level access-control system with tag-based policy specification and logic-based policy enforcement can, with reasonable efficiency, provide principled security for sharing personal files and metadata, using abstractions that fit well with users' desired access-control policies and preferences.

I validate this statement by (with colleagues) collecting and interpreting data about non-expert users' access-control policies and preferences; examining potential mechanisms for usable access control in depth via experience-sampling and lab studies; and designing, implementing and evaluating a prototype distributed file system called Penumbra that supports many of the access-control features identified by these studies as important for improving usability.

User policies and preferences. Using an in-situ, semi-structured interview study, colleagues and I developed a basic understanding of non-experts' access-control needs. We found that users have important data they want to protect, that their ideal policies require fine-grained and flexible policy mechanisms, that awareness and control of how content is accessed are important, and that current systems are not well aligned with users' mental models. These findings, along with a set of system-design goals derived from them,

provide the basis for establishing that Penumbra matches well with users' needs.

In two follow-up studies, we examined specific approaches to policymaking in more detail. We used an experience-sampling study to evaluate the potential of reactive policy creation, or an approach in which previously unauthorized access can be requested in near-real-time. We found that users have dynamic policies which reactive policy creation can potentially support, as well as that the approach has strong promise for providing the awareness and control lacking in current systems. In addition, we used a lab study to evaluate whether tag-based policy creation fits well with users' mental models for their ideal policies. We found that tags can effectively characterize the policies users want to implement for photos, and that users are comfortable with them, suggesting that this approach also has strong promise for helping users manage access-control more precisely and easily. Penumbra supports tag-based policy creation directly, and Penumbra's logic-based approach to policy enforcement can provide a foundation for supporting reactive policy creation.

Principled security. Penumbra is the first file-system access-control architecture that combines semantic policy specification with logic-based credentials. This approach combines support for intuitive, flexible policies with strong correctness. Penumbra's design supports independent access control for files and tags, distributed file access, private tags, tag disagreement between users, decentralized policy enforcement, and unforgeable audit records that describe who accessed what content and why that access was allowed.

Penumbra's logic can express a variety of flexible policies that map well to real users' needs. To demonstrate this, we developed a set of synthetic case studies, drawn directly from our user studies, that demonstrate that users' desired polices can be encoded effectively for our system.

Reasonable efficiency. We validate that Penumbra operates with reasonable efficiency using several performance tests. Using file-access benchmarks created from the synthetic case studies mentioned above, we measure Penumbra's latency for system calls and compare it to a control case in which all accesses succeed without requiring proofs. We find that for most system calls, the median latency is well below the 100 ms threshold identified in the HCI literature as a human-visible delay [100], and the overhead introduced by our proving system is less than 5%. In addition, we use several microbenchmarks to examine how proving time scales as different aspects of the policy space change. We find that proving scales well within and beyond the range of the small-group use cases we target, despite minimal optimization of our proof generation and caching techniques.

Thesis organization. This thesis contains eleven chapters. Chapter 2 presents related work covering a broad variety of connected areas. Chapter 3 describes the initial needs-assessment user study, undertaken to acquire insight into real users' access-control needs and preferences. Chapters 4 and 5 describe the follow-up studies examining in depth specific mechanisms for improving the usability of access control for personal data. Content from these chapters was previously published at CHI [78, 88, 89]. Chapter 6 presents the design of Penumbra, a distributed file system with access-control mechanisms inspired by the results of the three user studies, while Chapter 7 provides details of the access-control logic at the core of Penumbra. In Chapter 8, I detail the realistic case studies developed to support evaluation of Penumbra. Chapter 9 presents details of the implementation and evaluation of Penumbra. Much of the content in these chapters was previously published at FAST 2014 [90], but has been expanded to provide more detail about the Penumbra logic and extended evaluation. Chapter 10 synthesizes the lessons learned from our work in usable access control as a guide to future work, and Chapter 11 concludes.

2 | Related work

In this chapter, I discuss related work in four key areas: analysis of users' access-control needs and preferences; general approaches for improving the usability of access control; tagging in access control and in personal data management; and logic-based access control.

2.1 Access control policies and preferences

Considerable research, primarily in the HCI community, has addressed users' access-control policies and preferences in a variety of contexts. Prior to the user studies presented in Chapters 3-5, much of this work dealt with access control for professional, corporate, or educational environments rather than home or personal data. More recently, the increasing ubiquity of social networking has generated significant research on associated access-control needs. Across these varied contexts, several common themes have emerged. Although policies for corporate or professional data are more static than those for personal data, overall users' policy preferences are nuanced, dynamic, and context-dependent. The work presented in this thesis reinforces these ideas.

In this section, *ideal policy* refers to the policy a user would prefer if it could be implemented effortlessly.

2.1.1 Corporate and educational environments

Studies examining access-control policy in corporate and educational environments have found that while policies are sometimes simpler and more static than the policies for personal data described in this thesis, nuanced and complex policies can play an important role. Further, several studies report that users would hypothetically prefer finer-grained policies (if such policies were easily to implement), and that users do take advantage of more flexible options when available.

Olson et al. asked participants to indicate how comfortable they were sharing each of 40 types of personal information with each of 19 types of people, ranging from spouses and parents to bosses and company newsletters [102]. Using a hierarchical clustering algorithm,

they generated categories of people and information that were typically treated similarly. The results show that while some preferences can be captured using broad categories—for example, close friends and family members—some policies require fine-grained rules. Chapter 3 of this thesis demonstrates that the need for fine-grained rules is as or more important when dealing with personal files in a home environment.

Voida et al. studied how the affordances of various then-available file-sharing mechanisms affected sharing decisions and practices for 10 computer experts within a research organization and found that sharing decisions followed many of the patterns Olson et al. identified [136]. Additionally, they found that although email lacks many desirable sharing features, it remained the most popular sharing mechanism, due to a combination of universality and simple control. One important goal of this thesis is to better align system affordances with users' ideal sharing policies. Voida's participants also reported an important disconnect between mechanisms for content sharing itself and those for notification about updates; this push-pull distinction is also reflected in the findings of the photo-sharing study discussed in Chapter 5.

Using data mining, Smetters and Good examined how access controls were used in practice over a ten-year period in a corporate document-sharing system [124]. Similarly to Olson et al., they found that users explicitly set non-default policies on only a small portion of total documents, but that when they did the resulting policy tended to be complex. Once set, these policies were rarely updated. As in Voida's work, restrictions and usability difficulties in the existing policy-specification interface affected sharing decisions and frequently led to inefficiencies and errors.

The Grey system, deployed in an academic environment, provides configurable access control to both computer systems and physical office space [21]. Results from a nine-month deployment study demonstrated that when dynamic, fine-grained policies are explicitly enabled, users will take advantage of them to create and maintain nuanced policies that were not tenable using prior, less flexible systems (in this case, physical keys) [20]. This validates the prior findings from both Smetters and Voida that usability difficulties restrict the policies users apply in practice. These results fit well with findings in this thesis showing that users' ideal policies are complex, even when those users are unwilling or unable to implement them using current systems.

Razavi and Iverson investigated sharing preferences in a shared online personal-learning space [109]. Users expressed interest in fine-grained policies to help manage sharing a wide variety of content, different pieces of which had very different target audiences, especially when the content is captured for long-term archival. One of several specific factors affecting sharing decisions was content lifecycle, a finding echoed by the study reported in Chapter 4.

2.1.2 Social networks

In social networking applications, the literature suggests that while users want to protect personal content from strangers, they are perhaps more concerned about managing access and impressions among family, friends, and acquaintances. Managing access control within these types of relationships is a core focus of this thesis.

Ahern et al. examined privacy decision-making in the context of sharing photos and associated geodata on Flickr [8]. Among other results, the authors found that users wanted to hide unflattering content and restrict access to interests and activities that they prefer to keep private. Users also expressed concern about managing interpersonal relationships and group dynamics, such as preventing one friend from realizing they were excluded from a group activity. Similar findings about impression management were reported by Besmer and Lipford in a study of Facebook photo-tagging behavior and by Johnson et al. in a study of attitudes toward sharing content with Facebook friends [29, 66]. These concerns were echoed by participants in the user studies described in Chapters 3-5.

Policy limitations and coping mechanisms. As in the professional settings discussed above, implemented policies on social networks often deviate from ideal policies as a result of limitations in the system's policymaking affordances and/or to enable more convenient sharing. In 2006, Acquisti and Gross reported that a significant minority of Facebook users were concerned about privacy but either did not realize how much information they had made publicly available or did not realize that they could use privacy settings to reduce their exposure [5]. Ahern et al. similarly found that reported ideal photo sharing policies frequently did not match observed implemented policies [8].Relatedly, Johnson et al. and Wisniewski et al. report that social-network users apply ad-hoc mechanisms for managing disclosure, including curating friend networks, deleting content, employing multiple accounts, and even withdrawing from the network [66, 145]. These findings can be connected to similar ad-hoc approaches to protecting locally stored content reported in Chapter 3. Consistent with these results, a large-scale Facebook study by Staddon et al. demonstrates that users with higher privacy concerns are less engaged with the social network across several measures [127]. Wisniewski also notes a lack of in-band mechanisms to support negotiating or coordinating privacy boundaries directly; the reactive policy creation approach examined in Chapter 4 could partially address this gap.

Policy evolution over time. Two studies have considered how sharing preferences in social networks evolve over time. In a retrospective study, Ayalon and Toch found that users wanted the audience for Facebook posts to decrease slightly over time [15]. These results were explained in part by content losing relevance over time, by life changes, and by

changes in the participants' interpersonal relationships. Similar explanations for dynamic policy were provided by participants in the reactive policy creation study described in Chapter 4.

Bauer et al. also examined the temporal dimension of Facebook sharing, using a combination of retrospective and longitudinal studies [18]. In contrast to Ayalon and Toch's results, this study found that, on average, audience size remained relatively constant over time, with participants wanting some content to fade away and other content to remain more visible. Critically, study participants proved to be surprisingly poor at predicting how they would want the audience for any given post to change. This underscores the need to support policymaking in context rather than strictly a priori, a guideline discussed in Chapters 3 and 4 and emphasized in Chapter 10.

2.1.3 Other environments

Other studies have examined access-control preferences for personal data in other contexts, including home computing, device sharing, location sharing, and medical record sharing. As in Chapters 3-5 and in the research discussed above, nuanced ideal policies that account for contextual factors remain a common theme.

Two studies that consider specific aspects of home data sharing—device sharing and ubiquitous-computing-enabled shopping—find contrasting results about user preferences. In an in-situ interview study, Brush and Inkpen found that users sometimes need individualized accounts for shared computing devices in the home, but that personal accounts more frequently reflect a desire for personalization and organization than for privacy [34]. In contrast, a study examining reactions to a ubiquitous-computing shopping scenario—in which smart devices automatically generate and update shopping lists based on home activities—found considerably more concern about privacy within the home [84]. Participants worried that such a system could enable too much sharing about individuals' previously opaque habits and behaviors, promoting conflict within families.

While the Brush and Inkpen study considered shared-ownership devices within the home, Karlson et al. investigated users' preferences when lending their mobile phones, which are primarily single-user devices [72]. Participants indicated an interest in selecting among different levels of permitted activities when lending their phones. Sharing concerns varied meaningfully among participants, but often related not just to trust in the borrower but also to whether the owner is present. This finding about the importance of presence echoes one reported in Chapter 3.

An experience-sampling study of location-sharing preferences by Consolvo et al. also demonstrates the importance of context in policymaking [37]. Factors participants con-

sidered when deciding whether to share their location with a requester included the requester's location, the participant's location and activities, and even the participant's mood. In addition, sharing decisions were sometimes made in order to enforce social norms or send messages about inappropriate requests. The experience-sampling study of reactive policy creation presented in Chapter 4, which was partially based on this location-sharing study, found similar results in the context of sharing personal files.

A more recent study focused on the integration of location-sharing with social networks reported that users share their location not just to allow others to find them or coordinate meetings, but also for impression management and to maintain ties with distant friends or family [103]. Users were most likely to regret sharing location when the data spread outside its intended audience and across social contexts, leading to undesirable social consequences. The user study results presented in this thesis reflect similar concern about sensitive data unintentionally crossing social contexts.

Health data, too, exhibits patterns of nuanced policy preferences and variation among users similar to those found in Chapter 3 for broader categories of personal data. Caine and Hanania examined patients' preferences for sharing electronic medical records with potential recipients including primary and specialized medical providers, pharmacies, insurance companies, and family members [35]. Their results demonstrate that patients want fine-grained control over which aspects of their medical records are shared with which recipients. Further, the authors found important differences in sharing preferences between those patients with and without highly sensitive health information in their records.

Making matters even more complex, considerations of users' policy preferences must take cultural differences into account. Using online surveys as well as in-depth interviews, Ion et al. found important differences between Indian and Swiss cultures with respect to expectations of privacy for personal data stored in the cloud [65].

2.2 Improving usability for access control

The studies referenced in Section 2.1 indicate that users' desired policies are complex, dynamic, and varied. In a position paper, Ackerman argues that technology may never be able to accurately capture the nuanced social processes of everyday interpersonal sharing [4]. Nonetheless, many researchers have considered how to improve the usability of access control in a variety of contexts. This section reports on many such efforts, with the exception of efforts related to tagging (which are treated separately in Section 2.3.2).

2.2.1 Models and definitions for usable access control

Some researchers have attempted to define formal and informal models of access control usability. Cao and Iverson showed that implementing even simple desired policies in the WebDAV ACL system was conceptually and practically difficult for users [36]. As an alternative, they propose the *intentional access management* model, in which the user expressed his desired outcome, the system translates that into an implemented policy, and the system warns the user about unintended consequences, ambiguities, or conflicts that may result.

Beckerle and Martucci formally define six goals for usable access control, including that policies must allow and deny access to the correct resources to the correct people, that policies should have neither redundancies nor conflicts, and that the number of policy rules should be minimized [27]. From these definitions, as well as associated cost functions measuring the cost of security violations, the authors can optimize among policy rulesets for a given access control problem. Using user studies, the authors validated that their optimization criteria reflect IT experts' ranking of the usability of example policy rulesets for shared files.

2.2.2 Access control for files

Efforts to improve usability of access control for files in traditional operating systems include both improved visualization and improved interface semantics. Reeder et al. present the Expandable Grid, an interactive matrix for file-permission visualization in Windows XP [110]. This approach allows users to view their total policy at a high level, easily zoom in to inspect particular policy items in more detail, and modify policy directly from the visualization. Heitzmann et al. developed a tool to visualize hierarchical file-system policy using treemaps; this tool focuses on highlighting policy inheritance relationships among folder hierarchies [60]. Vaniea et al. report that displaying access-control policy information in close proximity to the content it controls helps users detect and fix policy misconfigurations [134]. A usable interface for Penumbra should incorporate best-practices for policy visualization and modification drawn from prior work in this area.

Johnson et al. define a *laissez-faire sharing* model that empowers end-users to consistently and transparently access files they create, delegate authority over those files, and enforce their chosen access-control policies dependably [68]. Penumbra's file-sharing model (see Chapter 6) is also designed to empower end-users and meets most of the goals of the laissez-faire model. Johnson et al. also discuss their attempt to retrofit the laissez-faire model on top of Windows shared folders, an attempt that was frustrated because the underlying system architecture was not aligned with users' file-sharing needs. This experience provides

further justification for designing access-control infrastructure from the ground up to match user needs, a primary goal of this thesis.

Krishnan et al. developed an alternative interface for traditional POSIX ACLs, presenting declarative rather than operational semantics [79]. Via a user study, the authors demonstrate that the declarative semantics allow users to set policy more accurately and faster than the traditional POSIX semantics.

2.2.3 Social networks and friend grouping

Because managing permissions individually for each person a user knows is generally untenable, particularly in social networks where users typically have hundreds of connections, placing friends and acquaintances into groups that share permissions is an important piece of making access control usable. Several researchers have considered how to make it easier to create, manage, and view such groups. Although Penumbra's design supports delegating policy to groups (Chapter 6), it does not include any mechanisms for helping users define groups. Mechanisms such as the ones described in the works below could be applied to add usability to Penumbra at the user-interface level.

Kelley et al. tested four different mechanisms for helping users build groups, and found that the method used influenced the resulting group characteristics [73]. In addition, the grouping process requires iteration and refinement to become reliable enough to support access-control policy. ReGroup supports Facebook group creation using interactive machine learning; the algorithm both suggests additional members for existing groups and suggests group characteristics that can help to filter the list of friends during the membership selection process [9]. Jones and O'Neill also developed a friend-clustering algorithm, in their case informed by grouping criteria identified in a card-sorting user study [69]. Adu-Oppong et al. and Danezis also suggest using machine learning to infer friend groups [6, 41].

Rather than explicitly defining groups, Fang and LeFevre present a privacy wizard that uses machine learning to decide which of a users' friends should have access to the user's profile data [48]. The classifier constructs a decision tree based on features like the social graph and information from friends' profiles, and the authors also present a prototype visualization of the produced decision tree. Policy inference such as that suggested by these works—particularly interactive inference such as that used by ReGroup—is an important recommendation of Chapter 10.

Once groups have been created (either by users or automatically), the PViz tool helps users visualize them using Venn diagrams, human-readable labels, and multiple levels of granularity [91]. Anwar and Fong present a tool to help users visualize their social graph

and then view their own profiles from the point of view of potential accessors [11]. This helps users to understand their current policies and identify misconfigurations.

2.2.4 Granting permission to mobile applications

Considerable work has examined the process of granting permissions to apps on mobile platforms. Several researchers have proposed new tools and mechanisms to support informed policymaking by users.

Access-control gadgets capture user intent by requiring mobile applications that want to access user-sensitive resources (like the camera or address book) to use a special API that checks permissions at time of use [115]. The API interprets authentic user actions (such as pushing a button to take a picture) as implicit permission. Howell and Schechter suggest a similar approach [64].

Thompson et al. suggest an alternative take on the implicit permission model; they suggest using strict resource attribution to allow users to identify and deal with misbehaving applictions [131]. This approach reflects earlier work defining optimistic access control, which assumes that most accesses are legitimate. All accesses are permitted, but are carefully logged [107]. This approach requires careful, consistent, and well-publicized external auditing to ensure users do not abuse it. Using machine learning and a real-world application-permission dataset, Liu et al. suggest that while users' privacy preferences related to mobile apps do vary widely, a small set of preconfigured privacy profiles could support most users' needs [85].

Other work aimed at helping users better control mobile app permissions includes making privacy information more salient in the display when users are choosing apps [74], offering users the ability to set runtime rather than only install-time policies [28, 98], providing descriptions and visualizations of how much and what types of data applications send over the network or share with third parties [16, 142], and crowdsourcing privacy protection settings [7]. Felt et al. argue that designers should use a combination of optimistic, implicit, runtime and install-time permission systems, choosing the most appropriate mechanism for a given situation [50].

2.2.5 Location sharing

Managing access-control policy for location-sharing applications has also been an active area of research. Tsai et al. explored using feedback to help users allay privacy concerns around location sharing [133]. Their results indicated that users who saw a log of past requests for their location were more comfortable sharing their location with friends and strangers, agreeing to share more often and for longer periods of time. In follow-up work,

Toch et al. developed Locaccino, a location-sharing tool for Facebook that supports complex sharing policies that can depend on who is requesting access, where the user to be located currently is, and what time of day the request is made [132]. They found that while many users made only a few simple rules, users who moved around more frequently and those who were willing to share their location more often made more complex and expressive policies. This suggests that users who get more benefit from the tool also have more need for nuanced privacy settings. Penumbra can support both simple access-control policies for basic users and nuanced policies for sophisticated users with complex needs.

Cranshaw et al. apply interactive policy learning to location sharing [40]. In this approach, the algorithm develops a suggested policy based on existing user data, makes the resulting policy suggestion available for users to edit, and incorporates any such edits into its model for suggesting future policy. Participants in the tagging study described in Chapter 5 expressed interest in a similar system for making policy about photo sharing.

2.3 Tagging

The user study presented in Section 5 examines the utility of tagging for defining access-control policy and finds that this approach has potential to help users more easily express their nuanced preferences. Building on these findings, the Penumbra prototype file system, presented in Chapter 6, uses tags as policy building blocks. To place this work in context, in this section I present related work discussing users' tagging behavior and motivation, other systems and studies that consider tagging for access control, and systems that use tags for file placement and management.

2.3.1 Tagging behavior and motivation

Penumbra's design assumes a combination of automated and user-generated tags. To place user-generated tags in context, it's important to understand users' tagging motivation and behavior outside the context of access control, a topic which has been studied extensively. Marshall provided an early classification of annotation behaviors and motivations in the context of personal annotation for hypertext [86]. Gupta et al. provide a survey of relevant research on social tagging that explores users' motivations, tag contents, and tag recommendation approaches [55].

In a qualitative study of photo sharing on Flickr, Ames and Naaman classified users' tagging behavior along two axes: organizational or communicative, and intended for oneself or for others [10]. Users in different quadrants of this taxonomy prioritize personal organization, memory cues, allowing their photos to be easily found by others, and signaling identity information to other users, respectively. A follow-up quantitative study

found that these tagging motivations influence the number of unique tags they create [101]. The tagging study in Chapter 5 adds access control as an additional dimension of tagging motivation; the qualitative results show interesting interplay with the other tagging motivations participants employed.

Addressing social tagging in general, rather than for photos only, Strohmeier et al. distinguish between users who *categorize* items using a high-level taxonomy that can support later browsing and those who *describe* items using open terminology for later search [129]. This distinction can be loosely analogized to the organizational/communicative axis discussed above. In follow-up work, Zubiaga et al. demonstrate that while descriptive tagging is superior for information retrieval, categorizers provide better inputs to automated content classifiers [153]. Improving automated classification of content is identified in Chapter 10 as a key requirement for enabling more usable access control at scale.

Sen et al. explored tagging behavior and vocabulary in the context of a movie-recommendation system [121]. Among other results, they find that many users do not want to view others' personal tags. Although this finding primarily addresses annoyance related to being overwhelmed with unhelpful information rather than privacy, it provides additional evidence for the importance of treating tags separately from the content they describe, the approach used by Penumbra and described in Section 6.2.1.

2.3.2 Tags for access control

As described in Chapter 6, Penumbra relies on tags to define access-control policies. Researchers have prototyped other tag-based access-control systems for specific contexts.

Au Yeung et al. implemented an access-control system for web photo albums, with policies defined using semantic-web RDF tags, but did not examine usability [14]. Unlike Penumbra, this system protects photos and tags together: anyone who can see the photo can see all tags. By taking advantage of the larger RDF ontology, this system supports reasoning over categories of tags, allowing for generalized policies (such as restricting photos taken in Europe based on tags like "Paris" or "Germany"). Participants in the tagging study described in Chapter 5 expressed interest in this kind of policy generalization.

De Coi et al. provide fine-grained access control for corporate resources, also using policies defined by RDF tags [42]. Like Penumbra, this system supports separate policies for protecting metadata and content. Unlike Penumbra, the system primarily deals with predefined tag ontologies, such as "to" and "from" fields for e-mails. Assuming that available attributes are typically common across users allows the authors to optimize policy evaluation by pre-computing a three-dimensional policy grid of users, tags, and resources. Similar precomputation could improve Penumbra's performance (see Section 9.3), but

would be complicated by the lack of structure imposed on Penumbra's tags.

De Cristofaro et al. developed a privacy scheme for microblogging services (such as Twitter) that allows a user to subscribe to content based on hashtag labels, without revealing which hashtags she is interested in [43]. This approach is in some ways the inverse of Penumbra; we allow tag creators to control who can read their tags, whereas this mechanism allows tag readers to protect their queries.

Attribute-based encryption. Sahai and Waters introduced attribute-based encryption, which allows users who share specified *attributes* (such as membership in a specific group) to decrypt and access protected resources [116]. Goyal et al. extended this concept to allow selective decryption of specific cyphertexts within a larger set (for example, individual entries in an audit log) based on attributes associated with that cyphertext [54]. Individual users (or more specifically, their keys) can decrypt only those cyphertexts with attributes that match the user's permissions.

One interesting practical implementation of attribute-based encryption is Staddon et al.'s system for protecting sensitive portions of legal and corporate documents [126]. The system can use either automatically generated tags produced by natural-language algorithms or user-generated tags to categorize sensitive content. Penumbra can be roughly analogized to this kind of attribute-based access-control system, except with policy enforced by reference monitors on individual files rather than enforced by specialized encryption on particular sensitive values within documents.

Usability for tag-based access control. Other than the study described in Chapter 5, research investigating the usability of tag-based access control is limited. Hart et al. conducted a role-playing study focusing on blogs [58]. Participants used existing tags attached to real blog posts to construct tag-based policy rules on behalf of the blogger for his simulated social network. The results showed that rules generated using tags were just as accurate and faster to construct than writing policy on a per-post basis, and that the number of tags per policy was manageably small. Our study, in contrast, focuses on photos and asks users to work with their own photos, tags, and policy preferences; our results our similarly encouraging for the usability of tag-based policy.

Policy prediction using tags. Other work at the intersection of tagging and access control has investigated using tags to predict or recommend policy settings. Vyas et al. developed APPGen to infer privacy policies for Web 2.0 content [137]. At initialization, users specify sensitivity values for topics of interest. Individual tags are assigned to broader topics using semantic similarity clustering; based on these inferences, the tool suggests policy for

new content based on tags associated with that content.

In follow-up work, Squicciarini et al. developed A3P, a system for inferring privacy policy for images using automated image-content analysis as well as user-generated tags [125]. As with APPGen, users must supply a training set (in this case images and associated policies). New images are then classified based on their content and metadata, and associated policy is suggested. Users are shown the predicted policy and offered the option to accept it as-is or revise it.

Both APPGen and A3P were evaluated using user studies with semi-synthetic test scenarios, and both produced suggested policies that were largely deemed acceptable by participants. Automated policy recommendation systems like these could be leveraged to improve the usability of Penumbra; participants in the tagging study described in Chapter 5 indicated an interest in automated policy suggestion to make tag-based policies more usable. Policy suggestion systems generally are strongly recommended in Chapter 10 to help policy management scale for various kinds of personal data.

2.3.3 Tags for managing distributed personal files

Many distributed file systems use tags for file management, an idea introduced by Gifford et al. [53]. Many suggest tags will eclipse hierarchical management [120].

Several systems allow tag-based file management but do not explicitly provide access control. Ensemblue uses a central-server model and provides partial replication using persistent queries that offer limited semantic expression [105]. These queries can be used to customize application behavior and event notification for certain semantic classes of files. Anzere and Perspective both provide peer-to-peer sharing with replication explicitly governed by semantic policy [113, 117]. Perspective targets home environments and supports usable specification of replication policy. Anzere builds on this idea by incorporating resource management polices, such as acquiring and releasing potentially expensive cloud resources as other devices become available. In Penumbra, tags are used to query objects and to specify access-control policy (see Section 6.2.1); this could easily be extended to manage replication policy as well.

Eyo provides device transparency, meaning that users have a coherent view of their data regardless of which device they are using and whether that device is currently available [128]. To do this, Eyo treats metadata (including semantic tags) as a proxy for the content it describes. This requires that metadata items be treated as first-class objects, decoupled from the content they describe; Penumbra uses a similar approach. Unlike Penumbra, however, Eyo requires that all metadata is replicated everywhere, precluding selective sharing of tags.

Other systems do connect tag-based management with access control. Homeviews provides capability-based access control, but remote files are read-only and each capability governs files local to one device [52]. In contrast, Penumbra provides more principled policy enforcement and supports policies that apply across devices. Cimbiosys offers partial replication based on tag filtering, governed by fixed hierarchical access-control policies [147]. Our work (Chapter 3) indicates personal policies do not follow this fixed hierarchical model; Penumbra's more flexible logic builds policies around non-hierarchical, editable tags, and does not require a centralized trusted authority.

Lekakis et al. observe that in personal distributed file systems, unless access control is implemented in the replication layer, information will leak as files and metadata are exchanged [81]. Similarly to our approach, they suggest using name-value pairs to inform both replication and security; they combine this with a role-based access-control scheme. The mechanism for specifying policy in terms of metadata is not detailed.

2.4 Logic-based access control

Logic-based access control has a 20-year history in the security and privacy literature. By formalizing authorization mechanics, researchers hope to increase trust in system implementations. Much of the prior work deals with formalizing and analyzing authorization logics and languages; less frequently, the concepts of logic-based access-control are applied in realistic contexts. In this section, I highlight some of the most notable prior work in this area, as well as how Penumbra's logic (Chapter 7) builds on this tradition.

An early example of logic-based access control is Taos, which mapped authentication requests to proofs [146]. In Taos, logic is used to formally explain and reason about authentication and authorization, but the logic is not directly implemented in the system, as it is in some of the later work (including Penumbra). In related work, Abadi et al. provide a formal structure for a logic-based distributed access-control architecture, which has provided a foundation for much of the resulting work in this area [3].

Proof-carrying authentication (PCA) [12], in which proofs are submitted together with requests, puts the burden of proof on the requester, allowing the reference monitor to maintain only a simple, efficient verifier. PCA has been applied in a variety of systems, including for web pages, for physical access control, and for supporting coordination among different public-key-infrastructure systems [21, 24, 82]. Penumbra adopts a proof-carrying approach in which requesters are responsible for expensive proof generation that reference monitors only validate.

Balfanz et al. developed Placeless Documents, an access-control infrastructure for distributed document-sharing that was one of the first implemented systems built on

formal authorizatio logic [17]. In this system, policy is created and delegated on a per-document basis, rather than across categories of files as in Penumbra. Like PCA, this system requires requesters to submit certificates that demonstrate the request's validity; unlike many later systems (including Penumbra), the mechanism for selecting which certificates to submit is unspecified.

Perhaps most similar to Penumbra is PCFS, in which Garg and Fenning apply PCA to a local file system [51]. PCFS uses parameterized proof caching to allow fast verification of access while supporting time-of-use verification of system-state conditions, while Penumbra considers the efficiency of proof generation. PCFS is evaluated in part using a case study based on government policy for classified data. In contrast, Penumbra supports a wider, more flexible set of distributed policies targeting personal data.

Many different logical authorization languages based have been proposed, with properties that vary based on the targeted scenario. The Penumbra logic defined in Chapter 7 supports many features similar to those enabled by these languages, using them in an applied context. Binder was the earliest of these languages to support distributed policy [44]. RT and Cassandra both use Datalog to encode role-based access control; RT includes constructs for separation of duty and for requiring combined authorization by k of n principals, while Cassandra allows the selection of a constraint domain that can provide more or less expressiveness depending on the requirements of a particular system [26, 83]. Penumbra can implement role-based policies (via its support for user groups) as well as more expressive policies that include tags.

SecPAL, also based on Datalog with constraints, targets a natural-language-like syntax [25]. Like Penumbra, SecPAL targets usability; users are expected to use SecPAL directly, rather than through an abstracted interface like the one envisioned for Penumbra. The DKAL language is expressed in fixed-point logic rather than Datalog, allowing for increased expressiveness; DKAL also supports directed communication, allowing principals to make statements that can only be heard by the intended recipient [56]. This feature can be used to protect sensitive policy, an access-control goal discussed in Chapter 10.

The Nexus authorization logic (NAL) allows policy specification over arbitrary predicates [118]. Similarly to PCFS, NAL allows interpreted predicates that depend on current system state. This type of predicate could be used to support some of the interesting policy preferences expressed by users in the needs-assessment study discussed in Chapter 3, for example by using sensors to verify the presence of content owners before allowing access.

A related but different approach to logic-based access control is the Policymaker-Keynote-STRONGMAN family of work, in which policies are expressed as programs that are evaluated over supplied credentials to validate access [32, 49, 76].

One important benefit shared by most of these approaches to logic-based access con-

trol (including Penumbra) is meaningful auditing; logging proofs provides unforgeable evidence of which policy credentials were used to allow access. This can be used to reduce the trusted computing base, to assign blame for unintended accesses, and to help users detect and fix policy misconfigurations [135].

3 | Preliminary needs assessment

When I began my work, not much was known about how people think about and interact with access control in the home [31]. In particular, there was a lack of data about how much or what kind of access control would be required to allow home-centered data-sharing systems to be usable while providing the protections users want.

As a first step, I conducted (jointly with colleagues[1]) a preliminary needs-assessment study consisting of in-situ, semi-structured interviews with 33 non-technical computer users in 15 households. This study broadly examined current access-control attitudes, needs, and practices; in addition, we employed hypothetical scenarios to ask users about what their needs and preferences might be in a world where sharing digital files is routine and ubiquitous.

This study led to four key findings. First, we found that people construct a variety of ad-hoc access-control mechanisms, but that these do not entirely allay their concerns about protecting sensitive data. Second, we found that people's ideal access-control policies can be complicated; they are not always defined exclusively in standard role-based terms but can also incorporate factors like who is present, where the access occurs, and what device is being used. Third, we found that a-priori policy specification is often insufficient, because it does not align well with social models of politeness and permission. In addition, many participants expressed a desire to update their policies iteratively in reaction to data access requests. Fourth, we found that people's mental models of access control and of computer security in general are often misaligned with current system designs in ways that could leave users vulnerable.

In the rest of this chapter, I describe the study methodology, overview our participants, discuss each of the four finding in detail, and then provide a set of associated guidelines for designing usable access-control systems for digital data in the home environment.

3.1 Methodology

We gathered data using semi-structured, in-situ interviews to increase understanding of how home users think about controlling access to their digital files. In order to address a

[1] J.P. Arsenault, Joanna Bresee, Nitin Gupta, Iulia Ion, Christina Johns, Daniel Lee, Yuan Liang, Jenny Olsen,

broad sweep of possible scenarios, we decided to include any devices that reside at least part time in the home as well as any files that currently are or someday may be stored on these devices. We also considered possible access by anyone who might be able to use or connect to any of these devices, now or in the future: friends, family, colleagues, and even strangers. We did not start out with any hypothesis; instead, as we conducted and analyzed interviews iteratively, we developed theories about home users' preferences, needs, and mental models for access control.

3.1.1 Interview protocol

We conducted semi-structured, in-home interviews. We interviewed household members first as a group and then individually. All interviews were structured around a predetermined set of questions designed to cover a wide range of access-control-related topics. The questions were intended to encourage participants to discuss their past experiences along with their current behaviors, thoughts, and feelings. If a participant mentioned an interesting topic not in the questions, the interviewer probed further, but otherwise kept to the question list. At least two interviewers attended each session. The interviews were recorded, resulting in more than 30 hours of videotape.

Group interviews lasted approximately 30 minutes each and included all available household members. In these sessions, we asked participants how they currently protect important information both on paper and digitally. To guide participants' thinking, we asked them to draw maps of their homes and illustrate which devices and rooms they considered to be public or private. We also asked about current formal and informal policies for who can use which devices under which circumstances.

Individual interviews lasted 30 minutes to 1.5 hours per participant. The goal of the individual interviews was to understand how participants define their ideal access-control policies and what features they would find useful to implement desired policies. The interview protocol had three major components. First, we asked participants to describe past experiences when they were concerned that others might view or modify data in an unwanted way. This section was used to focus participants on why and when access-control policies would be important to them. Second, we used their home map to walk participants through a list of the types of digital data they own, asking generally about which types are more private and which types are more public. This list was used to guide the rest of the interview. Third, we asked participants to imagine that their data could be available to anyone, from any device, anywhere, and at any time. We presented ten

scenarios in order to learn whether and to what degree participants would find various dimensions of access-policy definition useful. The scenarios tested policy-specification factors including: the identity and location of the accessor, the device used for access, whether the file owner is present during the access, the time of day access is attempted, the location of the file owner, and the incidence of social events. We also asked participants to consider policy-management mechanisms including privacy indicators, a detailed access log, and *reactive policy creation*.[2]

We prompted participants with specific events and people in an attempt to discern their general attitudes toward specific access-control mechanisms. For example, in order to assess whether participants wanted to restrict assess based on person, we picked two people that the participant had mentioned—a close friend or family member and someone they were not close with—and asked: "Imagine _____ could view all of your files and data. What would you not want them to see or change?" We gauged the strength of these preferences by asking participants how upsetting a violation would be, using a five-point Likert scale ranging from 'don't care' (1) to 'devastating' (5). We noted the scenarios that resonated with participants and elicited strong examples.

3.1.2 Data analysis

As we completed early interviews, we recorded whether each participant was very interested, somewhat interested, or uninterested in each axis of control, along with high-level explanations of their responses. This course-grained analysis revealed interesting patterns that helped focus later sessions.

Once interviews were complete, we iteratively coded each interview. The first round of coding was collaborative, with team members working together to validate each other's results. We transcribed all the videos and then applied a more detailed process of topic and analytic coding [111, 112]. A single coder recorded fine-grained codes for each aspect of participants' answers in a searchable database designed for easy cross-referencing of participants and topics. As new concepts emerged, the coder revisited previously analyzed transcripts to see how the new concepts related. Team members worked together to group individual codes into increasingly larger consensus categories. Using this process, we were able to formulate broader theories about participants' access-control concerns, needs, and preferences.

Our results are purely qualitative. We sometimes report the number of participants

[2]Reactive policy creation allows a file or resource owner to make a semi-real-time policy adjustment in response to an attempted access that cannot otherwise succeed. In prior work related to physical access control, a system incorporating reactive policy creation enabled users to construct policies closer to their ideal policies than a system of physical keys [19].

who fall into a given category to provide context; this is not intended to imply statistical or quantitative significance. We have selected the most interesting or salient quotes and anecdotes to embody each point; these examples are meant to be illustrative but not exhaustive. All the results that we report are substantiated by data from several interviewees.

3.2 Participants

We recruited Pittsburgh-area participant households through Craigslist posts, emails to university distribution lists, and flyers hung at grocery stores and distributed to families at children's soccer games. Households were prescreened to include those with a range of digital devices storing at least a moderate amount of personal data, but to exclude households that included computer programmers. We interviewed 33 people, ranging from elementary-school students to retirees. Participant households, which included five couples, six families with children, and four sets of roommates, were each paid $50.

In this chapter, we refer to participants using a naming scheme that identifies their household type (C for couples, R for roommates, or F for families), household number within that type, and member letter. For example, participant R2A belongs to the second roommate household (R2) and is the first member of that household to be interviewed (A). Table 3.1 lists some demographic information about the participants.

Households had different numbers of devices, ranging from seven devices for four people (F5) to 29 devices for four people (F2). The most common devices included laptop and desktop computers, mobile phones, music players, DVRs, external storage devices, digital cameras, and video game systems. Many computers had passwords, and some had separate login accounts; PINs were also used on a few phones and music players and one DVR (for parental control), but not other types of devices. Overall, about one third of devices were primarily shared, while the rest were primarily for individual use. In general, couples and families shared more devices than roommates, but specific ways of sharing varied significantly.

3.3 People need access control

Unsurprisingly, we observed that people have data they consider important or sensitive and want to ensure this data is protected. We discuss this result here for completeness, as well as to shed light on specific concerns participants raised and on the sometimes-surprising ways in which they accomplish their access-control goals. In the first subsection, we demonstrate that people have data they classify as sensitive, and they find the idea of unauthorized people accessing this data disturbing. Next, we provide evidence that these

Household	Size	Interviewees	Laptop	Desktop	Storage	Phone	mp3	DVR	Dig. camera	Other
C1	2	28m banking supervisor; 26f Spanish teacher	2			2				2
C2	2	27m forensic toxicologist; 28f healthcare recruiter	1	1		2	1	1	2	4
C3	2	27m student; 27f research assistant	1	1	1	2	2		2	1
C4	2	53f, 59m retired		1		2			1	
C5	2	27m police academy student; 24f nurse	1	1	1	2	2		2	1
F1	3	30m driver; 33f research associate	1			2	1	1	1	2
F2	4	53f pastor; 22f student	5	2	11	5	5		1	
F3	5	43f accountant; 9m student	4	1		2	2	1	2	
F4	3	50f admin. assistant; 18m, 15m students	1	1		3	2	2	1	3
F5	4	44m curator; 43f printer; 12f, 9f students		1	3	1	2			
F6	4	36m systems analyst	3	2	2	2	2	2	2	
R1	7	20f, 21f, 20f students	7		8	4	4		4	1
R2	2	25f, 26f students	2		4	2	2		2	
R3	4	20m student	4		2	3	2		2	1
R4	3	18f student; 24m video producer; 22m salesman	3	2	5	3	2		2	2

Table 3.1: Participant demographics for the needs-assessment study. The columns identify household code; number of part- or full-time occupants; age, gender, and occupation of each interviewee; and number of household devices.

concerns are not just hypothetical; several participants described incidents where their data was put at risk or exposed. Finally, we discuss ways that people construct their own ad-hoc access-control mechanisms using both technical tools and social norms.

3.3.1 People have data they classify as sensitive

Almost all participants want to limit access to their personal data. When we asked participants to imagine a breach of their ideal policy, we found preferences for access limitations are often very strong. Eighteen participants across 13 households classified at least one hypothetical policy violation as a 4 or 5 on our Likert scale. These devastating or near-devastating scenarios included unauthorized access (read, modify, or delete) to financial data, schoolwork, email, hobby or activity files, work files, text messages, photos, home videos, journals, and home musical recordings.

Many participants considered unauthorized access by strangers, acquaintances, bosses, and teachers to be highly undesirable. Perhaps more surprisingly, several were equally disturbed by situations involving closer relationships like parents, children, family, friends, and even significant others.

Examples of these critical violations (along with their Likert scores) include F4A's children seeing her finances (4); C1B's boss seeing her photos (4); R4A's boyfriend seeing her personal documents or work files (4) or modifying them (5); and nine-year-old F5D's friends seeing her email (4).

3.3.2 People's concerns are not just hypothetical

Our results reveal that not only do people have data they want to protect, but their current protection mechanisms are sometimes inadequate (or perceived to be inadequate). We asked participants to recall situations where they were concerned their sensitive data might be at risk, as well as situations where their data was accessed improperly. Twenty-two participants could recall specific instances of concern; only six reported they had never had such concerns. Nine participants reported actual policy breaches of varying severity.

Participant F4A, a divorced mother of two teenage boys, reported concern about her sons accessing her email when she leaves her account logged in on a family computer. "Maybe someone sort of emails you a sexy email, or something, and I wouldn't want the kids to see it." R4B was upset when he caught a roommate in his bedroom, using his computer without permission. F2B said her roommate sometimes grabs her phone and looks through pictures on it without asking, which is "kind of uncomfortable." R4C has also had private photos exposed on more than one occasion, including one incident where his girlfriend "stumbled upon an ice skating video of me and my ex. And it wasn't

anything, but it was an awkward moment."

R2A, a law student, once lent her computer to her adolescent sister, who inserted random words into a class assignment. R2A turned in the altered paper without noticing and had to apologize to the professor. Participant F1A reported a less serious instance of data modification: his wife accidentally deleting shows from their DVR before he watched them. "It's frustrating, because you're expecting to see it.... But what can you do, it's already done, it's gone."

3.3.3 People use a variety of access-control mechanisms

Because people are concerned about limiting access to their sensitive files, they take precautionary measures to reduce the risk of exposure. We found that while some people use standard access-control tools, others have developed ad-hoc procedures. These include using both technical and social mechanisms whose actual efficacy may vary, but which participants find reassuring. In total, 30 of 33 participants, including at least one in every household, reported using precautionary measures.

Use accounts, passwords, and encryption. Seven participants use passwords, encryption, or separate accounts for access control. Four said they are careful to log out or lock the computer when they walk away. R4C said, "I guess I'm a security junkie with my phone. Encrypting my text messages, it's not really necessary. But it makes me feel comfortable." Like most participants who used passwords, R4A protects her laptop rather than individual files. She said she uses the password "just in case when we have guests over, that nobody thinks that, 'Well, it doesn't have a password, that means I can use it.' Just to better my chances of not having my identity or secret information taken."

Limit physical access to devices. In most participants' configurations, data boundaries are device boundaries; anyone using a device has access to all the data stored on it. As a result, many participants are cautious about lending their devices to others, even for tasks like checking email or browsing the web. Most participants allow only people they trust to access their devices. As 15-year-old F4C said, "Obviously I don't let anyone who walks through the door on to my computer, but if someone's on my computer I trust them." A few participants allow others to use their devices only if they are present to supervise, and another few don't allow it at all. In households R1 and R3, devices left in common areas are considered available for general use, while devices stored in individual bedrooms are considered private. Some participants in these and other households shut down or put away devices to discourage others from using them. One participant keeps her most important data on an external hard drive, which she physically hides from her roommates.

Hide sensitive files. Participants also attempt to hide files within the file system: A few

name sensitive files obscurely for concealment and others bury them in layers of directories. According to R2A, "If you name something '8F2R349,' who's going to look at that?" C2B said, "[My husband] is a good hider of things.... If someone was trying to find something specific and he had it hidden, it would take them a while."

Delete sensitive data. Six participants have deleted sensitive files to prevent others from seeing them. F1A has deleted pictures of his two-year-old daughter from his cell phone for this reason: "If I didn't want everyone to see them, I just had them for a little while and then I just deleted them." A few participants have closed Facebook accounts because of privacy concerns.

3.4 People need fine-grained access control

In practice, many current access-control systems designed for home users favor simple, coarse-grained access policies. In Windows XP, the default "My Documents" and "Shared" folders divide a user's files into those accessible only to her and those accessible to everyone on her network. Although more fine-grained controls are available, they may not be sufficiently usable, as evidenced by participants' attempts to hide files. Apple's iTunes offers options for sharing the user's entire library, sharing only selected playlists, and requiring a password for the shared files. This configuration does not allow users to share different subsets of music with different people. Facebook supplies rich, customizable access controls for photo albums, but there is no differentiation between reading and writing. Any user who can view a photo can also tag it and leave comments on it. HomeViews, which is designed to enable easy data sharing for home users, is limited to read-only access [52].

Our results indicate people's policy preferences may be incompatible with coarse-grained control mechanisms in several ways: 1) Some participants' policies include fine-grained divisions of people and files. 2) Dimensions of policy specification beyond person and file are also important in some circumstances. 3) Even when individual policies are relatively simple, they differ significantly across participants; there is no small set of default policies that could meet most people's needs completely. In the following subsections, we discuss each of these complicating factors.

3.4.1 Fine-grained division of people and files

Early in the individual interviews, we asked participants to explain which people they would allow to access which files. We found that many participants specified complex groupings for both dimensions.

For some participants, policy specification required many categories of photos, some of which had fuzzy boundaries. C5B, for example, made several kinds of distinctions among her photos. In her first attempt to categorize her photos, she divided them into photos she was willing to publish and those she wasn't. After further thought, she divided the restricted photos into four categories: truly private photos as well as separate groups to share with family, sorority sisters, and general friends. Even these distinctions did not prove entirely adequate—there were some pictures she might only want to share with the people pictured. She also said her boyfriend could see some of the truly private photos, but not others, particularly those involving ex-boyfriends. R4C had a similarly complex division of photos into overlapping categories; he also mentioned photos that should carry different restrictions even though they were taken at the same event. Currently, both of these participants manage photo sharing by over-restricting; if they don't feel they can control access to a photo precisely enough, they decline to share it at all. The need for multiple policy divisions is not unique to photos; other participants specified similar distinctions within categories like music, videos, school files, and work files.

Our results also indicate people, like files, cannot be easily divided into just a few groups. Popular person designations included significant other, friends, family, co-workers, and strangers, but these groups often required additional subdivision. Several participants differentiated policy for one or two "best" friends; others made distinctions among close friends, casual friends, and acquaintances. Within families, policy varied for siblings, parents, and children. R2A said she is "far more willing to show my sister things than my parents." Participants also make distinctions between bosses and colleagues as well as within groups of colleagues. C5A even differentiated among strangers: "I think I would feel less embarrassed if I knew someone 100 miles away was looking at it [a sensitive file] rather than someone on the bus."

Figure 3.1a summarizes one participant's ideal policy, indicating which files she would share (white), restrict (black), or sometimes share (gray) with which people. As this fairly-typical policy makes clear, access decisions are not binary across people or file types. The presence of gray squares indicates a finer level of detail would be required to completely specify this policy.

3.4.2 Dimensions beyond person and file

Factors beyond the person requesting access and the file being accessed also inform participants' ideal policies. We asked participants to think about differences between read and write permissions, as well as whether or not the participant was present during the access, the participant's location, the location of the accessor, the device used for access,

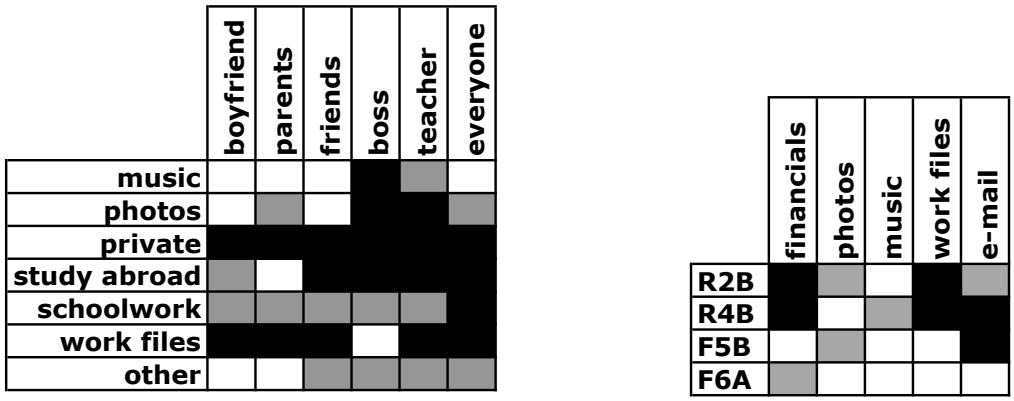

Figure 3.1: Variation within and among participants' ideal policies. (a) A high-level view of participant R4A's ideal policy. (b) Four participants' ideal policies. White squares indicate willingness to share; black squares indicate restriction; and gray squares indicate a willingness to share some files under some circumstances.

and the time of day of the access. Each of these factors was meaningful to at least a few of the participants.

Distinguishing read access from write access. Many participants described important policy differences between read access and modify/delete access. F4C said no one else should ever be able to modify any of his files; C2A and F1B were willing to grant their bosses only read access to some files. A few participants described general categories of files they were not concerned about sharing, but that they would want to protect from modification or deletion, including music, game files, schoolwork, and photos. This read-write distinction extends to highly trusted people such as family members and significant others. In one of several examples, middle-school-student F5C was willing to share almost all of her files with her family members, but did not want to grant modify or delete permissions. Similarly, R4A was willing to share highly sensitive files such as financial information and photos with her boyfriend, but did not want to grant him write access to any files.

On the other hand, a read-only system would not be sufficient for some participants, who see value in allowing others to edit their files sometimes. C5A wanted to let his mother improve his resume, and F2B would allow friends to provide feedback on scholarship essays. F5B would let clients update business files they send her. R2B expressed interest in allowing collaborators to edit files related to joint projects.

Presence. Policy specification based in part on whether or not the file owner is present resonated with a majority of participants. Participants believed that being present would allow them to exercise additional control over who accessed what, as well as providing

social pressure to encourage good behavior. F3B, a nine-year-old boy, said, "If I was next to [my friend], I would know which files he would be bringing up, but away from him I wouldn't have a clue what he was doing on my computer." R4C said, "If you have your mother in the room, you are not going to do anything bad. But if your mom is outside the room you can sneak." According to C3A, "If I'm in the house then it's likely that I'm spending time with them. If I'm not with them, I can find them and say, 'Hey! What are you doing on my computer?"'

For a few participants, being present provides additional benefits. Three said being present would allow them to make a last-minute decision to share something. R1B said she "might not remember what some of those files are," but if she is present, she can look at them and make an informed decision. Others said they wanted to be there to witness the accessor's reaction, explain things, and correct any misunderstanding. F6A wanted to make sure some opinionated journal entries wouldn't be misunderstood: "I could explain myself! Totally! If only I had a crystal ball for all the times somebody got upset with me and I didn't know it. If only I could have been there, then I could have told them: No, I am a lover, not a hater!" He also mentioned being present to explain things to his children: "Most movies I want to be there with [my son] ... in case he has questions or it's too scary. I can calm him down."

Location. We asked participants how location—their own or the file accessor's—would affect their ideal policies. A slight majority said they felt safer sharing data in their home than in other environments. C5A said, "I don't want them to look at my emails or texts. But if they were here, I wouldn't care if they wanted to look at my email. I don't know why, but I just feel more comfortable doing these things at home than being out in public with my information." Eight participants did not want to share any files in public places like buses or coffee shops. According to F1A, "Chang[ing] the settings as I move? That makes sense.... Going to work with the laptop vs. being at home—you might put it on extreme lockdown." In general, participants' responses to this question reflected their ideas about who was likely to be at a given location. R1C said, "At studio [at school] I am more hesitant to share my files if I am not there. In the apartment I can trust them with music or movie files. There is a mutual trust with people you live together with."

To many people, the accessor's location could be a proxy for trust: guests in the participants' homes were presumed to be trusted. Several participants said they would share more with people in their house; a few others would share more with people who were in their bedrooms, an even higher marker of likely trust. According to C2A, "I feel that if they are in my house I can control them a little more. If they are in their house, they have a freedom to do whatever they want and there is not a chance of me walking in on them."

On the other hand, some participants said their own location would not make a difference. According to F2B, "If there's a way to have a certain setting for a specific individual and have that setting not change based on location, then I wouldn't mind having the same access rights for my friends when I'm home or at school." Policies based on the location of the accessor also didn't make sense to many participants. As F6A said, "Just because you are inside my house, I would not categorize my files differently than if you were not there."

Device. We also asked participants whether or not the device used for access would affect their ideal policy. Most said the device had no effect, but a sizable minority did find it meaningful. To several participants, including R4A, devices with smaller screens are preferable for accessing sensitive files, as "it feels more private on a smaller screen." In contrast, others worried that a private device like a mobile phone might promote sneakier behavior than a public device like a television. According to F1B, "Maybe it's worse doing it on the laptop [than the TV], because of being a bit more private about it."

Time of day. We asked participants if their policies would vary according to the time of day when access was attempted. To a large majority, this idea did not make much sense; as C2B said, "It doesn't matter the time of day.... The things that I don't want you to see, I don't want you to see at any time. And no time would be worse than another time." A few, however, did find this possibility interesting. Some saw time of day as a proxy for presence or awareness; they did not want to share files while they were sleeping, because they could not know about or control the transaction. Said C3A, "If it's bedtime and I'm in bed, then I don't really get to see what people are looking at if I wanted to." F3A wanted to restrict her young sons' access to files at night, when they are supposed to be asleep.

3.4.3 Policies vary across people and households

As we have shown, some individuals' ideal policies are complex. Even when our participants' policies were relatively simple, however, they rarely overlapped. Thus, no standard set of default rules can be expected to meet most people's needs. R2B wanted to tightly restrict financial and work files, was willing to share email and photos with most friends, and was not at all concerned about sharing music. R4B, by contrast, did not consider his photos private but was concerned about sharing email and music, even with friends. F5B was interested in restricting email,but not concerned about sharing financial or work files. F6A did not consider anything except some financial information private. Figure 3.1b illustrates some of these variations.

Participants also had different attitudes about privacy. Many participants, including R2A, C4B, C5B, and R4C, started from the presumption that everything should be private

and then listed specific items to share with specific people. According to R2A, "Basically, it's my stuff; if I want you to have it I'll give it to you. If you want access to it, then ask." C5B said, "The files that I do have are private.... If I share it with you there's a specific reason." Other participants, including C3A, F4C, F5A, and F6A, started from the opposite position: sharing everything except a few specific exceptions. According to F5A, "I don't really have ... private files.... There's nothing that I am hiding from anybody." C3A said, "I'm not really that private. There's not a whole lot of stuff that I really want to keep from people aside from financial stuff."

In many cases, we found broad agreement on a general principle but enough variation in the details to make defining a satisfactory default policy difficult. For example, most participants identified one or two most trusted people—often a best friend or a spouse—to receive the most access. Within this group, about half were willing to grant this closest person complete access to everything; the other half wanted to grant access to most things but restrict access to some things. The specific exceptions varied and included everything from email, photos, and text messages to financial documents, work files, and even web-browser history.

3.5 Awareness and control

In the previous section, we showed that participants responded positively to several policy dimensions beyond person and file, including location, presence, and device being used. Participants gravitated toward options they perceived as providing the most visibility into and control over accesses to their files. As C5B explained, "I guess I'm not a terribly private person, but I think if someone's going to be meddling in your things, you should be able to know what exactly they're looking at."

While it is not surprising that participants are looking for more control over their data, their ideas about what control means and how to achieve it show unexpected variety and depth. In the following subsections, we describe three specific manifestations of this desire for control: a preference for being asked, a need for iterative policy creation and refinement, and an interest in knowing not just who is accessing their data but why.

3.5.1 Permission and control

Participants often think of digital data sharing in terms of asking and granting permission. For some, setting policy a priori does not feel the same as granting permission. We found that mechanisms such as being physically present and responding to system-prompted access requests can provide a stronger sense of permission-based control and therefore increase people's comfort with data sharing.

Many participants wanted an access-control mechanism that reflected standard social conventions of asking before using someone else's things. Three participants said no one should access anything without their express permission. C2B said, "In general I want to be asked. I'd prefer to give [my files] to them. I would not want someone to just look at them." C3A agreed, saying "I'm very willing to be open with people, I think I'd just like the courtesy of someone asking me.... If you ask someone nicely for pretty much anything, people will be more than willing to help you out." When we asked R4B about who is allowed to use his devices, he answered, "Any friend of mine who asks." According to C4A, "Without my permission, without my directly sending it to you, I wouldn't like you to look at ... the financial files or my email. That's my personal stuff."

To many of these participants, the idea of specifying in advance which people can access which files does not seem to convey a sufficient sense of control, possibly because they don't understand the idea of policy specification or they don't trust that policy will be enforced correctly. Several participants, when asked to describe their ideal policies, responded that no one should be able to access their files without their permission. We asked R2A what her boss should be restricted from seeing, and she responded, "Ideally I wouldn't want him to see anything except what I give him access to." Along the same lines, as discussed previously, several participants expressed concern about allowing access while they were sleeping. As R3A said, "I can't be giving you permission while I sleep because I am sleeping." Responses like these suggest that, to many participants, setting an access policy does not seem equivalent to granting permission.

Five people said that when they are present they can control which files can be accessed. According to C2A, "We don't get company that much, and we are usually constantly with our company. If they were viewing something, I would be there at all times, guiding them through where they should go or not." C1B said being present would affect her policy, "because I could say, 'These are the things that you could see."'

Participants responded positively to the idea of a reactive policy-creation system in part because they felt it would extend social conventions of permission into the digital world. C4A said a reactive system "sounds like the best possible scenario.... It would make me feel much more comfortable if people asked before they could modify or view the files at all. I like that a lot." Others said they would use a reactive system even for files they expected to rarely or never grant access to. C5A was open to making his financial documents—designated as highly restricted—available via such a system. "I don't think I would mind, if it asked me permission first. Say if an employer needs to see it.... I can't imagine too many people want to look at my stuff."

3.5.2 Iterative policy creation

For many participants, one important aspect of controlling access to their data was the ability to fine-tune policy easily and repeatedly. Some participants said they might want to make decisions about access at the last minute. R2B said she might change her policy "if there is something particularly relevant that I wanted to show, that I wouldn't normally want everyone to have unfettered access to." As discussed above, participants like C5A were interested in using a reactive policy-creation system to grant permission even to files they had not previously planned to share.

Three people placed particular emphasis on the ability to review policy and remove access. C2B said, "I would like to be able to go back on there and say, 'You said yes to all these people to view these things,' but if for some reason I no longer want them to do that, I could say 'denied' now and take them off the list."

Participants were also interested in fine-tuning their policies based on observed activities. Nine participants were interested in checking a detailed access log for unexpected or undesirable access patterns and then changing policy accordingly. C1A said, "It's nice to know who is accessing data more frequently. It opens the question: Are they the only ones viewing them, or are there other people standing next to them?" R1A added, "If someone has been looking at something a lot, I am going to be a little suspicious. In general, I would [then] restrict access to that specific file."

We also found evidence that at least some ideal policies change over time. C2B wanted to temporarily limit her sister's access when they fight. "She's not talking to us right now.... She's one of those people who, if you get mad at her, ... she'll rip up all the pictures of you. ... She could erase stuff on my computer."

3.5.3 Not just who, but why and for what purpose

Participants wanted to know not only who was accessing their files, but also why. C4B said, "Before you even touched anything, I would have to find out why you're doing it." F2A said she would like to use a reactive policy-creation system "if I know the purpose" for the request. F4B said a reactive policy-creation system "would be very useful, especially if maybe when they sent that they could add a message as to why they needed to see it." This was especially true for write permission—C5B said that she might grant permission to modify a file, "but I think I'd probably have to get into contact with them and ask them why they wanted to." C3B agreed: "I guess maybe if I got notified, like having the option of rejecting any of the changes, then that would be OK. Like if somebody changed it and it was better, then I could say OK, but if I didn't like it then I could reject it."

This interest extends to knowing how files will be used. F4B said, "I feel more comfort-

able if they're with me or I can see them, because then I have a better idea of what they're doing with whatever files they're seeing." He also mentioned a similar concern related to the device used for access: "If it was something portable, if they're using their phone, I might be worried about who else was watching." F5A felt more comfortable sharing files in his home, where he assumed it would be impossible to show files to an unauthorized third party without his noticing. F3A wanted to limit the devices used for access out of concern about people making copies of her files: "Probably I wouldn't want them to be able to save my information on their computers. 'Cause from my devices they would be able to view it but not save it."

3.6 Mental models and system designs don't match

Our interviews revealed several mismatches between people's mental models related to access control and current system designs and operations. These often occur because users carry assumptions from the physical world into the digital world, where they may be invalid or inadequately supported by system designers. These assumptions affect the ad-hoc access-control mechanisms people create as well as the factors that make them feel secure.

Hiding files in the file system. As we discussed, some participants attempt to hide sensitive files. This idea comes from from physical-world practices of hiding important items or mislabeling file folders to avoid suspicion. The couple in household C4, for example, keeps their most important papers in a small, hidden box; only less important papers are kept in the file cabinet, which is used as a decoy. The increasing availability of search tools, like Spotlight, Windows Search, and Google Desktop, that allow fast, accurate discovery of desired content regardless of file name or directory structure may invalidate this approach.

Preventing violations with presence. Based on physical-world experiences, many users believe being physically present can prevent policy violations. R4A, for example, said, "When I let people use my laptop, I'm usually near them, because it makes me feel comfortable that if anything were to happen, ... I'm right there to say, 'OK, what just happened?' So I'm not as worried." Participants note that their presence may increase social pressure against behaving badly. They also believe they will be able to notice policy violations and react quickly enough to prevent problems. Computer policy violations, however, are often faster or less obvious than physical-world break-ins, which may complicate detection even if the file owner is in the same room as the offender.

Device boundaries. Many participants base their access-control measures on the idea that device boundaries and data boundaries are the same—anyone using a device can access all the files on it and no files can be accessed without physically touching the device where

they are stored. As the increasing ubiquity of networking continues to blur distinctions between devices, this heuristic becomes less and less accurate. We also observed that users who subscribe to this model do not take advantage of tools like separate accounts or per-file encryption to segregate files within a device.

Location as a proxy. Some participants used the file accessor's location as a proxy for trust. For instance, based on the premise that only trusted people come into their homes, users would allow anyone within the home a high level of access to their data. It's not clear, however, that location is a particularly accurate proxy. C5B first said she would trust people in her house to access most files, but quickly changed her mind. "I guess originally my assumption would be ... if they were in the house, I'd know them, and they'd be close enough of a personal friend for them to actually be invited into my home. But then I was thinking, we've had plumbers here, guys laying carpet, stuff like that.... People are strange and might be snooping." In future work, it might be interesting to investigate whether the imprecision of this mechanism outweighs its convenience in real-world scenarios.

Infallible logs. Several participants wanted to use a detailed access log or notifications to verify enforcement of policy as well as to confront violators about their actions. F1A said, "[If] I all of a sudden got a thing [alert] on my phone, beep beep, somebody logged in to your account and is looking at it, yeah, I think that'd be great." According to C2A, a log would mean "I can call them on it [a violation] afterwards, and I would have proof of it." These statements rest on the assumption that even if the access-control system is sufficiently broken as to allow policy violations, the log or notification system would remain correct. This assumption seems dangerous, because an attacker sophisticated enough to bypass a reasonably robust access-control system may also be savvy enough to prevent her activities from being logged.

3.7 Guidelines for system designers

Based on our results, we have generated several guidelines for developers of access-control systems aimed at home users.

Allow fine-grained control. We found that participants' ideal policies were often complex and varied and were not always defined strictly in terms of files and people. It is important to keep in mind, however, that not all policies are fine-grained and not everyone wants to specify a detailed policy. An access-control interface should be designed to allow easy policy specification at multiple levels of granularity, according to the user's preference.

Plan for lending devices. We found that participants, especially those living with roommates, are often asked to lend computers to others who want to check e-mail or browse the web. Participants are often uncomfortable with these requests, because they worry

that the borrower will access private files or overwrite important data, either accidentally or on purpose. Karlson et al. suggest lightweight, limited-access guest profiles for mobile phones, with an emphasis on switching to this mode discreetly to avoid the appearance of distrust [72]. We suggest applying a similar approach to laptops and other devices.

Include reactive policy creation. Response to a hypothetical reactive policy-creation system was overwhelmingly positive, with 27 participants expressing interest in using such a system in at least some circumstances. R3A said, "I'd like that, it's useful. Only you can decide. That's something I would use." F4C answered, "That would be good.... Because then it would be easy access for them while still allowing me to control what they see."

Include logs. The majority of participants in our study also reacted positively to the idea of a detailed access log that would record all access attempts and their results. Some participants were interested in a log only out of curiosity, while others said that log contents might influence them to modify their policies. Six participants said they might share more if a log were available, including C4A, who said she would be "not a lot more open, but better than what I usually share." We recommend including a log or even a semi-real-time notification system designed to be human readable and to support policy changes based on log contents.

Reduce or eliminate up-front complexity. We found that although some participants' ideal policies are complex, defining fine-grained policies up front is difficult. Several participants, including C2A, reported that setting up a detailed access policy would be too much work. "If I had to sit down and sort everything into what people can view and cannot view, I think that would annoy me. I wouldn't do that." Even defining broad categories of access is seen as troublesome; participant R4C acknowledged he would not "go through the trouble of setting up a guest account" even to protect important files. As discussed earlier, some participants had difficulty specifying an ideal policy ahead of time and expressed interest in making last-minute policy decisions. We recommend reactive policy creation, either alone or in combination with preset policy, as one possible mechanism to reduce or even eliminate the up-front cost of setting fine-grained policies.

Acknowledge social conventions. A new design for an access-control system should take into account users' interest in the social convention of asking for permission. This is another instance where reactive policy creation could be helpful.

Another social convention for which we found strong interest was the idea of plausible deniability. Participants do not want to appear secretive or sneaky; as R4A said, "I don't want people to feel that I am hiding things from them." Several participants felt nervous about admitting they had private data and often felt compelled to justify it. C4A said, "Not that I have anything wrong or anything that can even be considered wrong, but I still want ... my privacy." Designers should take this into account and build into any new

system a means of unobtrusively restricting access.

Support iterative policy specification. We found that ideal policies change over time and users need to be able to easily review and refine their policies. We recommend creating an interface to allow users to see their current policy, review the resulting access logs, and make changes as needed.

Account for users' mental models. We discovered many instances where users' mental models of computer security in general and access control in particular are not well aligned with computer systems. New access-control systems should attempt either to fit into users' pre-existing mental models or to guide users to develop mental models consistent with the systems' behavior.

3.8 Summary

In this chapter, I described a broad study of non-expert users' general access-control needs and preferences. We found that although there is considerable variation in users' attitudes, goals, and mental models related to access control, some important themes did emerge. Most users have at least some sensitive or important content that they want or need to protect, and current approaches frequently do not match their mental models or effectively support their ideal policy goals.

Better addressing these goals will require supporting fine-grained divisions of people and files, as well as considering often-neglected decisionmaking factors such as who is present, where the access is taking place, and other situational circumstances. In general, users want a stronger sense of control over their data, including more acknowledgment of physical and social norms like asking permission or explaining why access is wanted. Based on these findings, we distill a set of guidelines for designers of systems for sharing personal content; these guidelines will be reflected in the design of the prototype system described in Chapter 6.

4 | Exploring reactive policy creation

In Chapter 4, I defined *reactive policy creation*, in which resource owners are not required to determine all access-control policies a priori, but may instead do so in response to a request. If a user tries to access a resource but lacks sufficient permission, she can use the access-control system to send a request to the resource owner, who can opt to update his policy and allow the access.

Ad-hoc reactive access control is frequently used in practice. When a person finds herself unable to access a given file, she may contact the file owner to ask permission. However, in most cases reactive policy creation is not supported directly by access-control systems. Instead, users go outside the system and make requests via e-mail or telephone calls. This disconnect between traditional system affordances and user behavior represents a potential opportunity for improving the user experience.

The results of the needs-assessment study suggested reactive policy creation could be a promising approach for non-technical home users; participants responded positively to the idea and indicated it could provide a greater sense of control over their data. In addition, it fit within the familiar social convention of asking for permission. However, that study asked participants to consider a reactive system only briefly.

To examine in more depth whether and how reactive policy creation could contribute to making access control more usable, I (jointly with colleagues[1]) designed and conducted an experience-sampling study. In particular, we wanted to know whether people have specific policy needs that match better to a reactive model than a traditional model, as well as whether reactive policy creation better matches users' mental models and preferences. We also wanted to know whether responding to requests would prove so tedious or annoying that the reactive model would be impractical. For this study, we chose to focus on how well the reactive model could work for file owners; we leave examining reactive policy creation from a requester's point of view to future work.

We collected a rich set of data that bolsters the case for using reactive policy creation as one of the modes by which home users specify file-access-control policy. We found

[1]Peter F. Klemperer, Richard Shay, Hassan Takabi, Lujo Bauer, and Lorrie Faith Cranor

quantitative and qualitative evidence of dynamic, situational access-control policies that are hard to implement using traditional models but that reactive policy creation can facilitate. Our study showed that the reactive model supports many of our participants' policy creation needs, including the desire for more control and interactivity.

While we found some clear disadvantages to the reactive model, they do not seem insurmountable. In fact, we found that some seemingly obvious disadvantages, such as the annoyance of receiving frequent requests, had only a minor impact on the usability of our simulated system and on user satisfaction. In the process, we captured detailed information about the policy decisions users made and how they made them. Our study also served as a very low-fidelity prototype, providing insight into effective interface design for incorporating reactive policy creation into the access-control system we plan to build.

In the rest of this chapter, I describe the study methodology, present the results, and discuss the limitations of our approach.

4.1 Methodology

This section describes our both experience-sampling and data analysis processes.

4.1.1 Experience-sampling process

We modeled our experience-sampling study on a location-sharing study by Consolvo et al. [37]. Our study included an initial briefing interview, a request phase, and a final debriefing interview. We used two conditions: *pre*-condition participants filled out a grid representing their proactive policy during the briefing interview, while *post*-condition participants filled out this grid during the debriefing interview.

Briefing interview. We collected lists of eight to 11 people with whom the participant might share files. Participants were required to list anyone they live with, a romantic partner if applicable, at least two family members and two friends not living with the participant, a supervisor, and at least two work colleagues or fellow students.

Participants were also asked to name files they store on digital devices. We prompted participants to think about photos, music, videos, financial files, work or school files, e-mail, and address-book information. In each category, we asked for groups of files, then one or two examples per group. We obtained 14 to 26 file names per participant.

Pre-condition participants only were then asked to define a *yes*, *no*, or *maybe* access policy for each of the file/requester combinations. We call this the *proactive* policy, because it is established before any hypothetical access attempts. The *maybe* policy indicates the participant cannot or does not want to make a decision without more information.

During the first round, participants appeared to have difficulty comprehending the *maybe* policy. To address this, we read a more detailed description of the *maybe* policy to second-round participants and asked them to provide and explain examples of *yes*, *no*, and *maybe* policy choices before completing the grid on their own. As we expected, this change led to increased use of *maybe* policies in the second round, mainly in exchange for decreased use of *no*. No other significant differences were found between the first and second rounds in the Likert questions, proactive grid policies, or reactive policies, except for a slight increase in participants who said they might use a reactive system to request files.[2] As a result, we consider the effect of conducting the study in two rounds to be small.

Request phase. In the second phase, participants received mock file-request e-mails indicating a particular person wanted to access a particular file. The people and files were randomly selected, using a uniform distribution, from the lists provided during the briefing; combinations were not repeated. Each participant received five to 15 requests per day between 7 a.m. and 11 p.m., during a six- to seven-day period, with the exact number and timing also randomly selected with uniform distribution. All requests were simulated; none of the participants' acquaintances were contacted, nor were their files actually shared. Requests were assigned randomly to examine a broad range of requester/file combinations, including those that participants might find strange or uncomfortable.

Each request directed the participant to a website where she could select a response from seven options, as illustrated in Figure 4.1. The options allowed the participant to *ignore*, *allow*, or *deny* the request. *Allow* and *deny* responses could be set for that request only, all future requests from that person for that file, or all future requests from that person for that file group. The participant was also asked to supply a short explanation of her reasoning, intended specifically for the researchers and not the hypothetical requester. In the second round, participants were also able to provide an optional free-form description of additional policies they would like to create.

Participants' persistent reactive policy decisions did not feed back into requests they received later. As a result, requests could be inconsistent with participants' previous policy decisions (e.g., a participant might receive a request from someone they had previously stated they wished to allow all requests from). A more complete system would have automatically screened out inconsistent requests.

We asked participants to imagine the requests were real when responding. We asked

[2] All differences calculated using unpaired t-tests. Significant p-values (<0.05): Likert *use to request*, p=0.0285; proactive *no*, p=0.0296; proactive *maybe*, p=0.000284. Non-significant p-values: Likert *convenience*, p=0.983; Likert *annoyance*, p=0.741; Likert *enjoyable*, p=0.272; Likert *use to share*, p=0.788; proactive *yes*, p=0.508; reactive *allow*, p=0.0884; reactive *deny*, p=0.125; reactive *ignore*, p=0.803.

them not to check their e-mail more often or respond sooner than they might if the requests were real, and to consider any factors that might influence their answer to a real request. During the debriefing interviews, we found that for the most part our participants immersed themselves in the system and demonstrated strong, sometimes emotional reactions to the simulated requests.

Participants responded to 1360 of 1452 total requests sent to them. The responses represented 30% coverage of 4481 possible requester/file combinations, with minimum coverage of 19% and maximum coverage of 45% per participant.

Debriefing interview. At the start of the debriefing interview, post-condition participants only were asked to define a proactive *yes*, *no*, or *maybe* access policy for all of the file/requester combinations they had defined in the briefing.

All participants then completed a seven-question survey. Using a seven-point Likert scale from 'strongly agree' (1) to 'strongly disagree' (7), we asked how enjoyable or convenient they found the system, and whether they would consider using such a system in real life. We also asked whether they found the number of messages they received annoying.

We then asked open-ended questions about their experience. We asked for which people and files they particularly did or did not want to use a reactive system, and how well the response options met their needs. We also asked how realistic they found the requests.

Then, we discussed in detail several individual requests, chosen to provide broad coverage and include particularly interesting cases. For each selected request, participants were asked to explain why they answered as they did and whether they would choose the same answer again. Where applicable, we asked about any social awkwardness that could arise from denying or ignoring a request.

Finally, we asked participants whether they would prefer to create policy all at once, up front; to use a reactive system; or to use a combination of the two, and why.

Grid conditions. As discussed earlier, participants were divided into two conditions. Participants in the first condition filled in a proactive access policy grid during the initial interview, while participants in the second condition completed the same grid during the debriefing. This grid allowed us to contrast the participants' proactive policies with the reactive policies created by responding to requests, and provided participants with a clear point of comparison between setting policy all at once and setting policy on a per-request basis. We used the conditions to examine priming effects on participants who were required to think through every possible person-file combination by filling out a grid before responding to requests.

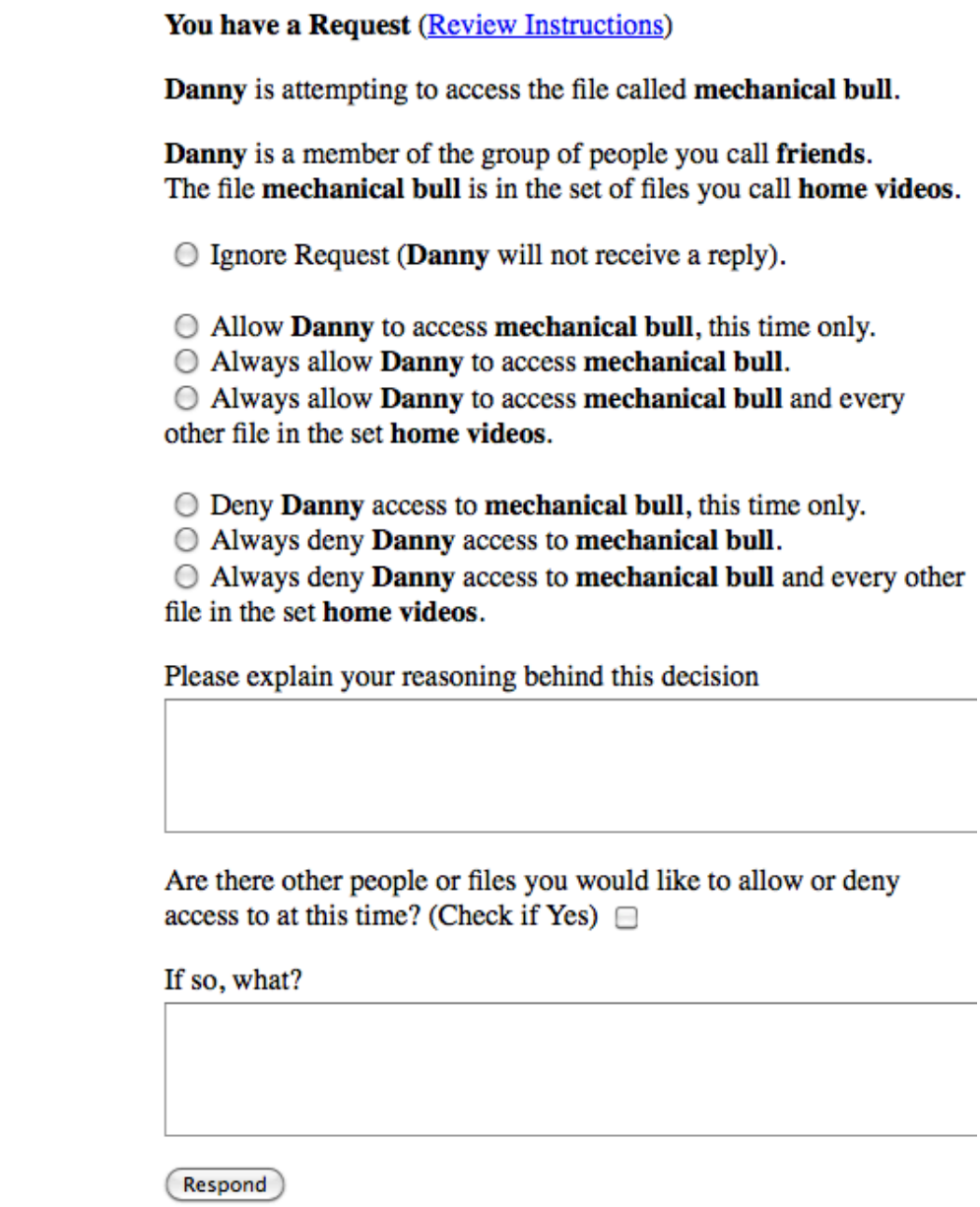

Figure 4.1: Sample response form.

In both cases, we emphasized to participants that we were not asking them to match their grid and request responses, and that changing their minds was normal and allowed. We cannot completely account for the fact that participants may have tried to match their answers to appear more consistent, but we attempted to minimize its effects.

4.1.2 Data analysis

We collected both quantitative and qualitative data. To evaluate participants' free-form responses to debriefing questions, we applied an iterative coding process [111, 112]. An initial fine-grained review of a subset of the debriefing survey responses resulted in an initial set of codes. Then two of the authors independently coded the participant responses. Disagreements were resolved through modification or clarification of the codes until consensus was reached.

Several participants provided no policy choice for some grid cells. To facilitate analysis, only files with complete grid policy definitions are included in this chapter. One participant failed to complete an entire page of grid policy, resulting in 97% of the blank grid entries. The dropped entries account for approximately 2% of possible grid policies.

A few participants answered the same request more than once, creating 26 duplicate responses (less than 2% of total responses). In this chapter, the first response provided by the participant for any requester/file combination is assumed correct and used for data analysis.

4.2 Findings

In this section, we first provide an overview of our participants and summarize the quantitative results. We then discuss seven specific findings drawn both from these results and from qualitative data that we collected. The first four provide evidence that users' policy needs can be better met by reactive policy creation than by traditional models. The next three describe additional interesting, somewhat unexpected qualitative results that can guide effective design of reactive-policy-creation systems.

4.2.1 Participants

We recruited 24 adult participants, using craigslist advertisements and fliers posted at our universities. We conducted the study in two rounds, one month apart, with 10 and 14 participants respectively. To focus on non-experts, we limited participation to those without degrees or jobs in computer science or engineering. Table 4.1 lists demographic information about our participants. Participants were compensated $10 for the initial briefing interview, $15 for the debriefing interview, and 25¢ for each response to a reactive request.

We encountered a large gender disparity during recruitment: the first 10 volunteers who met our requirements were all women. To counter this, we performed a second round of interviews with nine men and five women. While this may affect our results, we believe

Code	Age	Gender	Occupation	Household	Condition	Files	Requesters
P01	23	F	marketing manager	R	pre	24	10
P02	41	F	magazine editor	A	post	24	9
P03	26	F	law student	R	pre	20	8
P04	32	F	unemployed	C	post	22	10
P05	25	F	law student, legal secretary	A	post	20	11
P06	23	F	student	R	pre	19	10
P07	24	F	graduate student	A	post	20	9
P08	43	F	unemployed	A	pre	9	9
P09	29	F	student	R	pre	26	9
P10	46	F	video producer	C	post	25	9
P11	34	M	legal assistant	F	pre	26	8
P12	23	F	student	F	post	20	8
P13	23	F	student	C	pre	19	8
P14	22	F	student	R	pre	20	8
P15	37	F	business administrator	F	post	24	11
P16	23	F	product developer	C,R	post	25	8
P17	26	M	unemployed	R	pre	21	9
P18	37	M	HR manager	F	post	22	9
P19	34	M	lawyer	C	post	14	8
P20	23	M	marketing coordinator	C,R	pre	20	10
P21	54	M	purchasing manager	F	post	20	8
P22	21	M	student	R	pre	15	8
P23	24	M	bookkeeper	R	post	23	8
P24	26	M	entertainment	F	post	26	8

Table 4.1: Participant demographics for the reactive policy creation study. For household, R indicates roommates, F families, C couples, and A participants who live alone.

the effect is small. We discovered no major differences between the participants in the first and second round, apart from expected changes in the use of *maybe* policies, described below.

4.2.2 Overview of quantitative results

Participants filled in a total of 4481 policy grid entries. Of these, 56% (2518) were *yes*, 34% (1518) were *no*, and 10% (445) were *maybe*. Individual choices ranged from 100% *yes* (P22) to 72% *no* (P15). Details of each participant's grid policies can be found in Figure 4.2a.

Request responses showed a similar distribution, with 67% (913) *allow*, 30% (406) *deny*, and 3% (41) *ignore*. Individuals ranged from 100% *allow* (P22) to 62% *deny* (P01). Participants used *allow once* 251 times, *always allow file* 281 times, and *always allow group* 381 times. *Deny* responses were divided into 108 *deny once*, 81 *always deny file*, and 217 *always deny group*. Figure 4.2b summarizes individual participants' responses.

We also measured conflicts between participants' grid entries and their responses to

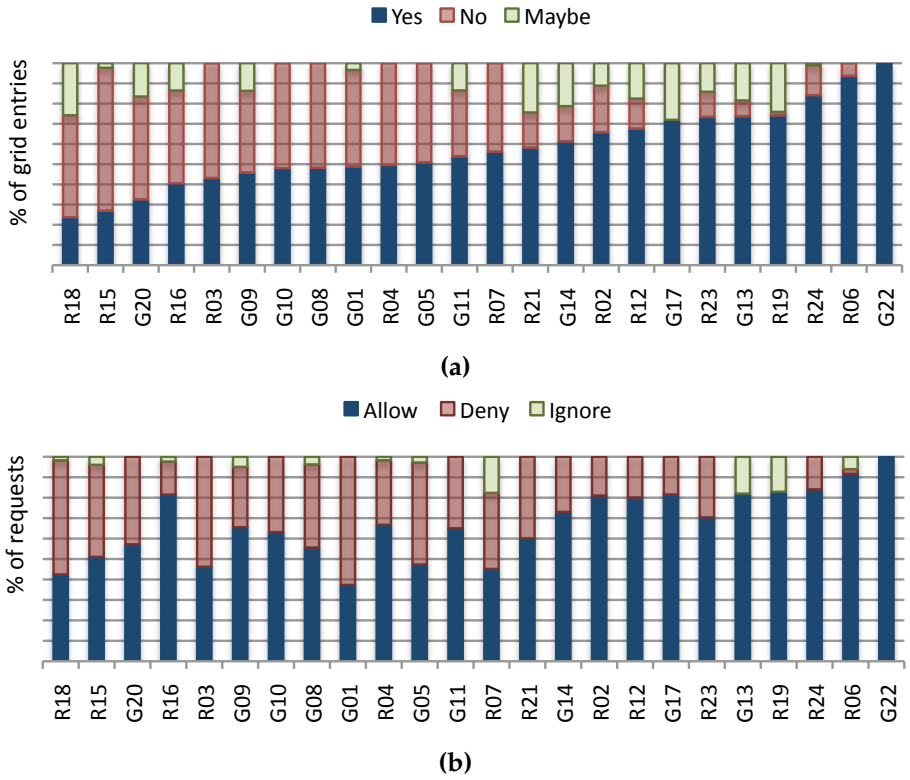

Figure 4.2: Variation between participants' proactive and reactive policies. (a) Participants' proactive policies, normalized as a percentage of that participant's possible requester-file combinations. (b) Participants' reactive policies, normalized as a percentage of requests to which the participant responded to the researchers. In both graphs, the x-axis is ordered by increasing percentage of *yes* grid entries.

requests. We consider instances where the participant marked *yes* but responded with any type of *deny*, or marked *no* but responded with any type of *allow*, as conflicts. Overall, 12% of responses resulted in conflicts, ranging from several participants with no conflicts to P08, for whom 49% of responses resulted in conflicts.

We performed a statistical analysis to more formally assess whether and how participants' sharing preferences changed between their grid entries and their reactive responses. Because our data includes multiple decisions from each participant, individual decisions are not independent. To account for this, we used the standard mixed-models approach, which treats each individual participant's decisions as a group. We treated both grid entries and reactive responses as ordinal outcomes, so we selected a *cumulative-link (logit) mixed model*, or CLMM [59]. For grid entries, we mapped *no*, *maybe*, and *yes* to 0, 1, and 2, respectively. For reactive responses, we mapped all *deny* variations to 0 (equivalent to *no*) and all *allow* variations to 2 (equivalent to *yes*). Deciding how to handle *ignore* was trickier,

as some participants considered it a middle ground between allow and deny (equivalent to *maybe*) while others treated it as a more socially acceptable way to deny access (equivalent to *no*, see Section 4.2.7). To account for this, we ran the model twice, once with *ignore* mapped each way, and applied Bonferroni correction to account for the multiple testing.

The results of the CLMM analysis show that regardless of how *ignore* was modeled, the distribution of sharing preferences did change significantly between participants' grid entries and reactive responses, with participants more likely to share in the reactive case ($p < 0.0002$ for each model). The CLMM likelihoods for reactive responses were 1.39 (*ignore* mapped to *maybe*) and 1.32 (*ignore* mapped to *no*). These values can be interpreted as indicating that for any decision and any ordinal sharing value j, oneof a participant's reactive responses is 1.39x or 1.32x as likely as one of her grid entries to have an ordinal sharing value greater than j. This increase of about a third shows that the reactive model has a powerful influence on user decision-making.

We found no significant difference in grid or response patterns between pre- and post-condition participants.[3]

During the debriefing interviews, participants answered several Likert questions about their experiences. When asked whether they enjoyed using the system, the median response was a moderately positive 5 out of 7 (66.7% agree, 16.7% disagree, 16.7% neutral). Participants also agreed (median 6 out of 7; 66.7% agree, 12.5% neutral, 20.8% disagree) that this type of system would be convenient for them. Asked whether they found the e-mail requests annoying, participants disagreed slightly (median 3 out of 7; 25% agree, 16.7% neutral, 58.3% disagree).

We also asked participants whether they would use a similar system in real life, both to share their own files and to request files from others. Eleven participants said they probably would use such a system, nine said they might or might not, and only four said they probably would not.

In addition, we asked participants whether they would prefer reactive policy specification (represented by the request system), proactive policy specification (represented by the grid they filled out), or a combination of the two. Seven participants preferred the reactive model and 14 preferred the combination; only one preferred proactive policy alone.

During the second round, we asked participants if they would rather receive requests via e-mail, text message, phone call or other means. They overwhelmingly preferred e-mail, often in combination with text messages. A few participants asked for requests to be aggregated, either in a periodic digest e-mail or via a web-based service the participant could check at her convenience.

[3]Differences calculated using unpaired t-tests, with p-values as follows: proactive *yes*, p=0.952; proactive *no*, p=0.972; proactive *maybe*, p=0.967; reactive *allow*, p=0.808; reactive *deny*, p=0.704; reactive *ignore*, p=0.619.

4.2.3 Policies change over time

Our results indicate that participants' file-sharing policies change relatively often, in response to a variety of factors. Reactive policy creation is better suited to express these kinds of dynamic policies than traditional access-control models. Participants demonstrated this dynamism through their proactive and reactive policy choices, including use of *maybe*, one-time decisions, and policy conflicts.

One indicator of dynamic policy was the frequent use of *maybe* in participants' policy grids, both before and after using the request system. As mentioned previously, in the second half of the study, when we clarified the *maybe* option, the use of *maybe* rose from 3% of grid entries in round one to 15% of grid entries in round two. Second-round participants used *maybe* more often at the expense of *no*; the rate of using *yes* stayed roughly unchanged. Especially in round two, participants often used *maybe* in thoughtful, deliberate ways in cases where context was important and access policy could not be entirely specified in advance. For example, P21 used *maybe* in reference to sharing financial data from his son's business with some of his co-workers. He never expected to share that file with those people, but said he might make an exception if an accountant wanted to provide financial advice. He chose *maybe* rather than *no*, planning to make an informed decision in context. P17 never used *no* in his grid, preferring *maybe* for the same reason.

As with *maybe*, participants used the one-time *allow* and *deny* options to express policies that were expected to change. P10 selected *deny once* rather than *always deny* for one request because "it is within the realm of possibility that something would alter" and she would change her mind. P23 denied a work colleague access to a photo of him with friends once, after marking that combination maybe in the grid. He explained that he might grant access "maybe if we were just having casual talk at work and I mentioned something we did over the weekend."

In many cases, differences between proactive and reactive policy also indicated changing policy preferences. We found that 12% of total request responses conflicted with the participants' grid entries. Although a few of these conflicts were generated by participant misclicks in the response form, most reflect actual policy changes. Table 4.2 details the conflicts we observed. Interestingly, 62% of all conflicts involved answering *no* in the grid but allowing a request, regardless of whether the participant filled out the grid first or answered the request first. This provides some evidence that people will share more reactively than proactively.

P05 approved an access on her grid, but refused the same access as a request because "today he's on my blacklist." P12 said, "Some files might be consistent all the time," while others "depend on [the requester's] reasoning or might depend on my mood." P02 refused

Proactive *yes*, reactive *deny*	
Number of yes-deny conflicts	64
As percent of all conflicts	38.1
As percent of all responses	4.7
Proactive *no*, reactive *allow*	
Number of no-allow conflicts	104
As percent of all conflicts	61.9
As percent of all responses	7.7
Total conflicts	
Number of conflicts	168
As percent of all responses	12.4

Table 4.2: Conflicts between proactive and reactive policy. Most conflicts occurred when a proactive *no* decision was overturned by a reactive *allow* decision.

a request for a work document in progress, but later marked yes in her grid because the document had since been completed. (This is consistent with Razavi's findings about sharing patterns over document life cycles [109].)

4.2.4 Policies are situational

Our results indicate that participants' policies are dynamic in part because their sharing decisions depend heavily on the details of the situation at the time the access-control decision is made. Again, this context-awareness is a natural fit for a reactive model, which allows users to make decisions at the relevant time rather than a priori.

Participants frequently explained that the reason why someone wanted to access a file mattered in making policy. P04 denied a request from a friend for video of a family wedding because "there's no reason she would want to see it. ... If it had come with some kind of explanation for a reason why," then she might have permitted the access. P10 said, "Almost every answer I have is based on context." She chose to allow her husband to see an invoice from her business only once, saying she would want a reason to give him the file because he is "less careful than I am about sharing digital information." P12 used *deny once* for several requests for videos of herself practicing public speaking because "I don't really like to be recorded and on camera, but in the future if there were some really good reason, I'd allow it." She also said she might consider a request from a professor for work from a different class, but she would want a valid academic reason to share it. P08 added that if she were sending requests, she would want to supply a reason, as it seems "presumptuous" to ask without explaining.

Some participants even invented reasons for our randomly generated requests. P23 allowed a request from a friend for his sister's contact information once only, suggesting he might accept that request in an emergency. P12 guessed a friend might want to see

her resume "as an example." P13 allowed a work colleague's request for a term paper she'd written on 'Feminism and Film' because "maybe he's interested in the topic and if I can help him get a broader understanding of it, then that would be good." She said an explanation of the request would help her make a better decision about a file like this one, which is "kind of personal but then kind of not."

Other examples demonstrate that this situational dependence can make it difficult to accurately specify policy ahead of time. Seeing a request helped remind P06 of ethical considerations. In reference to a request from another student for a term paper, she said, "Sharing your work with another student has potential to get you in trouble. ... At the time [when filling out the grid], I didn't think it would be an issue with me, but actually seeing, 'Matt is requesting your term paper,' the light went off, saying 'bad idea." In her grid, P01 allowed her work photographs to be seen by many requesters. However, when the requests were sent, she reconsidered her professional responsibilities and decided against sharing those files. P02 had the opposite reaction; she allowed a request for a sensitive file, but then said no in the grid because seeing all the people at once on the grid reminded her that information shared with one person will often be disseminated further.

4.2.5 Policies are also complex in other ways

Our findings indicate that many users' policy preferences are also complex in less dynamic ways.

Some participants considered factors beyond the sensitivity of the information in question when making decisions. For example, P09 denied a request from a friend for a Christmas photograph because "she doesn't celebrate Christmas and might be offended." P07 denied several requests for music when she thought the requester might not enjoy the song in question.

Two participants said they might like to grant fine-grained permissions to sections within files, not just to files themselves. P02 keeps all her passwords written down in one file; discussing a request from her teenage daughter for that file, she said that she might want to share some of those passwords with her daughter, but not others, "because someone who's younger doesn't know how to disseminate it or not." Similarly, P21 said he might want to share inventory and planning spreadsheets from his construction business with clients or vendors but redact some cost and pricing details.

4.2.6 Reactive policy creation fits users' interest in control

Several of our participants found a request-based system appealing because they felt it provided added control over the dissemination of their files. This finding confirms that

reactive policy creation continues to fit well into users' mental models after a week of simulated exposure to it.

Six participants said they might use reactive policy creation to help them track who was accessing their files and when. P11 said he would use a system like this one to "see who is actually accessing my files" and create considered responses. P21 liked that a system like this would provide a record of "who had access to what." P02 said a reactive policy creation model would make it easier than a proactive model to start saying no if someone is abusing access privileges. P02 also never used the group options for a response, saying that she wanted to know which individual files people were accessing, even if that would mean receiving more requests. P13 added that "it makes me feel comfortable knowing what people are trying to access."

Other participants liked that the reactive policy model incorporates the idea of requesting permission. P01 said, "I enjoyed people asking for permission to see the files." P16 said she used *maybe* for certain grid entries because "sometimes I would be willing to share ... but I'd like him to ask me."

Participants also said the reactive model helped them make better decisions. P15 said answering requests "made you think"; with current systems, she often sends files or forwards e-mails "automatically, without thinking." P14 said the reactive model provides "more of an opportunity to really think about it."

4.2.7 Social norms influence policy choices

As a low-fidelity prototype, our study provided insight into the ways people might use a reactive-policy-creation system. Social factors played a large role in participants' reactions, as well as in the specific policy decisions they made. Understanding these influences can help designers structure a reactive system to maximize user comfort.

Many participants expressed discomfort at receiving requests they considered inappropriate, such as from friends for confidential work documents or from co-workers for financial files. Several said they were confident they would never receive such requests from real people, who would "know better."

Participants had interesting reactions to the *ignore* option. The 11 participants who used this option applied it only 41 times, totaling only 3% of responses from all users. Seven participants told us they expected never to use this option, mainly because they found it rude not to send a reply. P01 said she wouldn't use *ignore* because she would like a reply if she sent requests. P05 said she wouldn't use *ignore* because "I don't like to live in the gray area." Others suggested that using *ignore* would only postpone the problem, as the requester would simply ask again until she received a response.

Some did see value in using *ignore*. Most commonly, *ignore* was used when the participant felt uncomfortable sharing the file but also uncomfortable denying the request outright. In many cases, this happened when a supervisor or authority figure asked for personal materials. For example, P13 felt "uncomfortable sharing personal pictures with a professor I am not close with, but I don't want to deny him access outright and make him feel uncomfortable. I figure ignoring his request will get the message across." She added that "For me, an ignore is like, 'I never want you to see it, but I don't want to talk about why.' It's just a more passive rejection for me." P06 used *ignore* to avoid saying no to her father, who had asked to see a video from her 21st birthday party that she preferred not to share with him. A few participants suggested that if they chose to *ignore* a requester, that requester might forget about the request entirely, neatly solving the problem.

Other explanations for using *ignore* included adding a delay while deciding how to answer, handling requests the participant considered too outlandish or inappropriate to deserve an answer, and handling requests for files the participant could not immediately identify. One participant used *ignore* to register her displeasure with a friend who had "made me upset that day."

We also asked participants directly if it bothered them to refuse or ignore requests, and if they worried that the requester would be upset. Most participants said they weren't bothered, saying that people who asked for inappropriate files should expect to receive a negative result. This is one area where we think the gap between experience sampling and real life has a strong impact; we expect that people might be more worried about the social consequences of denying or ignoring requests from real people. On the other hand, P06 pointed out that ignoring or denying a request is no worse within a reactive policy creation system than in any other sharing model, "just because I think eventually you have to do that anyway."

Social factors also played a role in several participants' desire to manage file access "manually," in person or otherwise outside a file-sharing system like the one we proposed. Several participants said they might not use a system like ours to request files, preferring to ask directly, over the phone or even by e-mail, rather than allowing the system to send an automated request. P07 said she would rather share in person: "I can show [this friend] this collection of music some other time when I see her on a daily basis." P10, a video producer, said she might use a request for one of her films to generate a personal interaction. She wanted to send a message with her response, saying "I'd love to share that with you. ... Do you want to watch together?" and hoping to "develop a conversation." P04 was also looking for more personal interaction, saying, "If my stepmother wants my friend's contact information, she needs to personally talk to me."

As discussed above, our participants were far more likely to say *yes* than *no* in their

grids, and even more likely to accept a request than to say *yes* in the grid. Some of this may be attributable to the files being self-selected (discussed in more detail later), but it may also reflect a general social pressure to say yes when asked for something and to avoid the appearance of having secrets. In addition, as mentioned above, most conflicts between proactive and reactive policy were more permissive reactively. Taken together, these results suggest that perhaps forcing users to respond to direct requests increases social pressure to share.

4.2.8 People have difficulty trusting systems

Several participants were concerned about the security of our proposed system. Some worried that a system that exposed files for access via requests could be attacked, allowing unauthorized access. P08 said she would be "sort of paranoid" about exposing her files this way, in case an attacker "somehow [got] in to take other stuff" or "plant a worm on your drive. ... I would feel more comfortable if I would send it versus they go in and get it off my computer." Another participant worried that showing the existence of financial files could provide incentive for attackers to attempt to break into her system. Others were concerned that the source of a request could not be properly authenticated, and worried they might grant permission to a malicious user masquerading as a friend or co-worker. P08 suggested a spammer would "sooner or later ... hit on a name that was someone I know," and P18 asked "can someone request a file [with my friends name] and I'm giving access that I shouldn't be?"

These concerns, expressed by one quarter of participants, arose organically; in our script, we were careful to specify that "no one can access [your files] without your permission," but participants were unwilling to take our word for it, even with no real data at risk. This suggests a strong level of skepticism toward new systems claiming to provide secure data sharing. Designers of new reactive systems must take this into account and find ways to convincingly demonstrate security in order to get user buy-in.

4.2.9 Reactive policy specification raises specific concerns

Although we found significant evidence of the need for and popularity of reactive policy creation, we also identified specific concerns that must be carefully managed for a system based on this model to be successful. These potential problem areas include annoyance with too many messages, the possibility that users may not recognize a requested file, the need for well-chosen response options, and the potential for disclosing information via lists of requestable files.

Annoyance with requests. We feared that even a small set of daily requests would cause annoyance and delay in answering, outweighing the possible benefits of reactive policy specification. We discovered this was not at all the case—participants expressed little annoyance and for the most part answered requests promptly. Most participants reported answering requests immediately upon receiving the e-mail. Measured request-response intervals accord with this, showing no increase in delay from the beginning to the end of the study. On a Likert scale, participants disagreed slightly with the statement "I found the e-mail requests annoying." We also asked them whether the number of requests they received was "way too many, very annoying"; "more than I would have preferred but not so many that it really bothered me"; or "a reasonable number of messages." Only one participant chose "way too many"; the rest split evenly between the other two options. Several participants indicated that although they received a lot of messages during the study, they would expect to receive many fewer in real life, limiting the annoyance. A few participants said the requests were no worse than the normal volume of e-mail they receive.

Although our ability to realistically evaluate the annoyance potential of a request-based system was somewhat limited, our impression after talking with participants is that while some frequency of requests would be intolerably annoying, our study did not reach that level. We are optimistic that a real system could maintain a tolerable message frequency. We expect that requests in a real system would decrease over time, as users grant persistent permissions or apply policy to groups of people or files and the system handles more and more situations automatically. This depends, of course, on how often users select the *always* and *always group* responses. In our study, participants selected these options more than two-thirds of the time, despite the fact that only a few coarse-grained file groups were available. P13 said she and a friend "share photos all the time," so she would allow all that friend to see all photos because it's "less of a hassle" than handling them individually. P23 also mentioned that he wished "the system after a while would have recognized" his previous choices and then sent fewer requests.

Trouble recognizing files. The reactive policy model assumes users can make effective decisions when presented with a request for file access. This assumption breaks down if the user does not recognize the file in question. At least three participants in our study could not identify a requested file, despite the fact that we used only a small subset of their files, which we asked them to describe memorably less than a week prior to the request. We expect this problem to get worse when dealing with a user's complete set of files, some of which may not be named descriptively, over an indeterminate period of time. This problem could be mitigated by offering the user a chance to view the file in question before

making a decision.

Additional response options. We asked participants if they had ever wanted to provide a response not available on the form. The most popular suggestion was to reply asking why the file was wanted, further demonstrating the context-dependent nature of some policies. Other interesting suggestions included a "request pending" response to provide assurance the request was not being ignored, as well as an option to forward the request directly to a third party to make a decision. The forwarding option was mentioned in the context of work-related requests requiring approval from a supervisor as well as requests for address book information that should be approved by the person whose contact information was being requested.

Information disclosure via lists of files. For a request-based system to work, requesters will likely need some information about what files or directories are available. This will by necessity leak information about what files exist, possibly including sensitive information if filenames are specific and descriptive. In a real system, we would expect to provide users with some control over file visibility; for simplicity, in this study we assigned requests randomly rather than allowing users discretion.

We asked participants how this issue would affect their interest in using this kind of system. Some participants weren't worried at all, because the files they wanted to restrict included things like tax files or calendar entries; these participants reasoned that "everyone" has files like those and so revealing their existence wouldn't be damaging.

As expected, however, many participants did express concerns. These concerns generally took two forms: worry that listing files would provide temptation for attackers to attempt to break the system security, and worry that sensitive information would be leaked as part of file and directory names. For example, P21 mentioned keeping the existence of a new resume file from a boss when searching for a new job; another participant was concerned that revealing e-mail subject lines might lead to a friend discovering a social event she hadn't been invited to. We believe these concerns can be effectively addressed by allowing users to decide which files should be visible and requestable to which people.

4.3 Limitations

It can be difficult to evaluate how well a proposed system feature will work without actually implementing the system to test it. Because of this, we put significant thought into our study design, and evaluating its success was one of our major goals. Despite the inherent limitations of experience-sampling simulation and the specific limitations of our

methodology, overall we found evidence to suggest that our results can apply to reactive policy-creation systems in general.

First, although we asked participants to imagine receiving requests and sharing their files, they were aware no data was actually at risk. As a result, a participant might have refused a simulated request that in reality she might have accepted in order to avoid an awkward social situation. On the other hand, participants might have more casually accepted simulated requests than they would real ones. This problem is compounded by randomly generated requests that sometimes appeared bizarre or inappropriate, combined with our inability to tell participants why a given request was made. Several participants said they might accept requests they considered unusual or inappropriate if the requester had a good reason, which we were unable to supply. This may have reduced participants' ability to imagine the system to be real. However, based on our debriefing interviews we contend that our participants suspended their disbelief, took the requests they received seriously, and answered carefully.

In addition, our reactive response form allowed participants to explain their decisions. Participants provided reasons for 87% of responses, indicating thoughtful decision making. We also asked each participant for details about several individual responses; their detailed, reflective answers demonstrate they took the requests seriously. Participants who made policy decisions because they were angry at the requester or concerned she would not like the content, or who created justification scenarios for unlikely requests, clearly engaged with the system as though it were real.

Our decision to pay participants 25¢ per response created another potential limitation. We paid participants this way partially in order to replicate social incentives to respond to requests received from friends, family, and colleagues. The payment, however, might have induced participants to respond to more requests than they would have otherwise, or else reduced their annoyance at receiving requests. Because the payment per request was so small, we don't believe it introduced very much skew into our results. In addition, during the debriefing we asked participants directly about the annoyance of receiving requests; their frank and thoughtful replies provide at least some evidence that annoyance was not suppressed by the payment. Participants' tendency to respond relatively quickly to requests (as discussed earlier) also suggests they were motivated more by interest than payment, as they were paid regardless of when they answered.

Another possible limitation is that the requestors and files used in our study were selected by the participants. Because participants selected only a small subset of their files, it is likely they chose not to mention some of the most sensitive or private items. They also selected only a small subset of the people they know, so it is likely that some people with whom they have unique or complex relationships were left out. We tried to mitigate this

by asking about a diverse variety of people and files, and by requiring each participant to supply at least one requester and file in each of several standard categories, including potentially sensitive categories like financial files and supervisors.

We asked participants to use our simulated reactive-policy-creation system for only a week. Longer use might cause behaviors and opinions to change. We believe, however, that the one-week period was enough to gain valuable insights. Observing most participants, it quickly became clear which requesters or files would result in strong policy preferences and which would be complex, dynamic, or borderline cases. Even dynamic decisions often followed similar lines of reasoning for each participant. As a result, we believe that for most participants we reached a saturation point where we had explored the majority of their policy decision space.

Our methodology only addressed reactive policy creation from the point of view of the resource owner; in future work, we hope to examine it from the requester's perspective. We also do not consider how users could view or modify existing policies created via *always* or *never* responses, which would be an important piece of a real reactive policy-creation system. A practical system would also need to consider how users could verify the source of a request.

4.4 Summary

In this chapter, I described a study designed to determine if users' expressed interest in a reactive policy creation tool would hold up under simulated use of such a tool. Despite some limitations, the experience-sampling methodology we used yielded rich quantitative and qualitative data about users' access-control decisions and the factors that influence them.

Reaction to our simulated system for reactive policy creation was encouraging, if not definitive. We found evidence of access-control policies that are hard to express using existing static mechanisms, but that reactive policy creation can facilitate. Of the policies we collected from participants in our study, 21% used *maybe* or involved conflicts, and hence could not be easily expressed without a reactive policy-creation mechanism or other extension to traditional policy-creation practices. An additional 16% of policies were one-time policies, meant to be changed after a single access. These too may be difficult to define using traditional methods, and we conjecture that many of them would benefit from reactive policy creation.

We also found that when making policy decisions, people want more control and interactivity and rely on social norms, all areas where reactive policy creation can contribute to an access-control system. We found that while there are some clear disadvantages to

the reactive model, they don't seem insurmountable. Overall, our results demonstrate that reactive policy creation is a strong candidate for further research and for potential inclusion in future access-control systems.

5 | Exploring tag-based policy for photos

Prior work has considered the applicability of tags to file management and access control [14, 58, 117]. However, to my knowledge the usability of tag-based access control had not previously been investigated using users' own content, tags, and policies.

In this chapter, I present an 18-participant lab study (conducted jointly with colleagues[1]), in which we use participants' own photos to explore the feasibility of tag-based access-control rules for photo sharing. Although tag-based access control could potentially apply to broader categories of digital content, we draw on photo sharing as an initial case study both because users have varied access-control preferences for photos and because systems that allow users to tag photos are already in use.

To explore the efficacy of tag-based access-control rules, the study sought to answer the following research questions:

- ***Q1: Can organizational tags be repurposed as-is to create effective access-control rules?***: Users already create tags for purposes including organization, search, description, and communication. To allow tag-based access control to function with minimal overhead, can rules based on these currently available tags capture user preferences?

- ***Q2: Does tagging with access control in mind improve the performance of tag-based access control?***: For tag-based access control to be practical, users must intuit how adding and removing tags affects their access-control policies. When tagging with access control in mind, do users' tags more accurately capture their preferences?

- ***Q3: How do users engage with the concept of tag-based access control?***: Tag-based rule creation should be intuitive and understandable for users. What strategies do users employ when simultaneously tagging photos for both their current, organizational purposes and access control? What do users understand and like about tag-based access control, and what impediments does tag-based access control

[1] Peter F. Klemperer, Yuan Liang, Manya Sleeper, Blase Ur, Lujo Bauer, Lorrie Faith Cranor, Nitin Gupta, and Michael K. Reiter

present? Can users understand and suggest tag-based access-control rules that support their preferences?

We found that organizational tags could be repurposed to create efficient and reasonably accurate access-control rules. When participants tagged photos with access control in mind, they were typically able to develop coherent strategies and create tags that supported significantly more accurate rules than those created from organizational tags alone. We also observed that participants understood the concept of tag-based rules and were able to actively engage in rule suggestion.

In the rest of this chapter, I detail the study methodology, then present basic data about our participants' demographics, access-control policies, and tags. I then proceed to our main results and analysis, followed by a discussion of study limitations. I conclude by highlighting implications for the design of tag-based access-control systems.

5.1 Methodology

We designed an exploratory laboratory study during which participants performed three separate tagging tasks. The first task focused exclusively on *organizational tagging* to help a user organize and search her photos, while the second and third tasks focused on organizational tagging in combination with tagging for access control. These tasks provided insight into participants' tagging behaviors and strategies (Q3). Tags from these tasks were also used to create machine-generated access-control rules that roughly approximated users' policies. Some of these rules were shown to the participants to demonstrate the tag-based access-control concept and stimulate discussion (Q3). We also used the tags and machine-generated rules during post-processing to evaluate the efficacy of organizational tags for access control (Q1) and to compare the performance of organizational-only tags to combined organizational and access-control tags (Q2).

5.1.1 Recruitment

We used advertisements on Craigslist, in the university's newspaper, on a university research participant website, and on posters around Pittsburgh to recruit English-speaking participants who take at least 100 photos per year. Because we were interested in the usefulness of existing organizational tagging strategies for access control, we required that participants add keyword tags or captions to photos "often" or "always." We eliminated participants who only tagged photos on Flickr or Facebook since Flickr tags tend to be created for sharing [94], and Facebook tags are limited to people's names. We felt that including such users would skew the results, although excluding them may limit

generalizability.

Qualified participants were asked to upload 40 photos they had previously tagged. To prompt potential participants to provide photos for which they might have varied access preferences, we provided them with a list of 17 suggested photo categories, including "up to 15 photos that you haven't posted publicly and wouldn't want to share with the general public," "3 photos with trees in them," and "3 photos with your relatives in them." We also asked them whether they would be willing to share the photos with "some," "none," or "all" of several groups of people. Potential participants who answered "none" or "all" to all categories were eliminated. As we discuss in detail in the Limitations section, we believe these measures were successful.

We sent our screening survey to 152 people, 63 of whom completed the survey. Of those 63, we rejected 39 people who only tagged photos on Facebook or Flickr, 5 who did not tag frequently enough, and 1 who lacked English proficiency. The remaining 18 people made up our participants.

5.1.2 Procedures

Qualified participants were invited to our lab for the main part of the study, which lasted between 1.5 and 2.5 hours. During this portion of the study, we observed their organizational tagging behaviors and their strategies for incorporating access control into their tagging schemes. We also presented participants with machine-generated, tag-based access-control rules, both to demonstrate tag-based rules and to gauge their reactions. For this portion of the study, we used the Picasa desktop photo software and custom web interfaces.

Warmup task. We gave each participant a brief tutorial on tagging photos in Picasa. As a warmup task, we asked her to add at least one tag to each of five sample photos unrelated to her own photos.

T1: Organizational tagging. T1 was designed to evaluate how effective organizational tags can be for expressing access-control policies.

Prior to the lab session, we stripped all existing tags from the participant's photos, saving these *original tags* for later reference. In the lab, we asked the participant to re-tag her stripped photos with the objective of finding the photos more easily in the future, adding as many tags as she would like. We asked participants to re-tag their photos so we could observe each participant's tagging behavior using a think-aloud procedure.

Collecting access-control preferences. Next, we collected a set of the participant's access-control preferences for her study photos. These preferences served as ground truth for creating and evaluating access-control rules.

We collected a list of people with whom the participant might want to share photos. We first prompted for three people with whom she had a close relationship, including household members, friends, and significant others. We then prompted for four to seven people with whom she had less close relationships, such as supervisors, friends of friends, neighbors, and colleagues. From here forward, we will refer to this entire set of people as the participant's *friends.*

We then presented the participant with an *access grid* mapping her photos to her friends. We added a "Public" friend column to represent posting a photo publicly, in connection with the participant's real name. For each combination of friend and photo, we asked the participant to select a preference from the following options:

- **Strong allow**: Allow access; would be upset if the friend were not able to view the photo.
- **Weak allow**: Allow access; would not be very upset if the friend were not able to view the photo.
- **Strong deny**: Deny access; would be upset if the friend were able to view the photo.
- **Weak deny**: Deny access; would not be very upset if the friend were able to view the photo.
- **Neutral**: Absolutely no preference between allowing and denying the friend access to the photo.

To confirm understanding, we asked the participant to point out and explain one example for each type of preference (or to explain why that preference would not be needed).

Example rules. At this point, we introduced the participant to the concept of tag-based access-control rules. To aid this introduction, we created a set of machine-generated best-fit access-control rules for each of the participant's friends. Rule generation is described in more detail below. Each friend was assigned zero or more rules of the form "If *tagged / not tagged* with *tag*, then *allow / deny*," combined with *and* and *or* as appropriate. Each friend was also assigned a *default rule* of allow or deny, which applied to any photos not covered by the other rules. Some friends were assigned only default rules—for example, the participant's spouse might be assigned the simple rule of "always allow" access to all

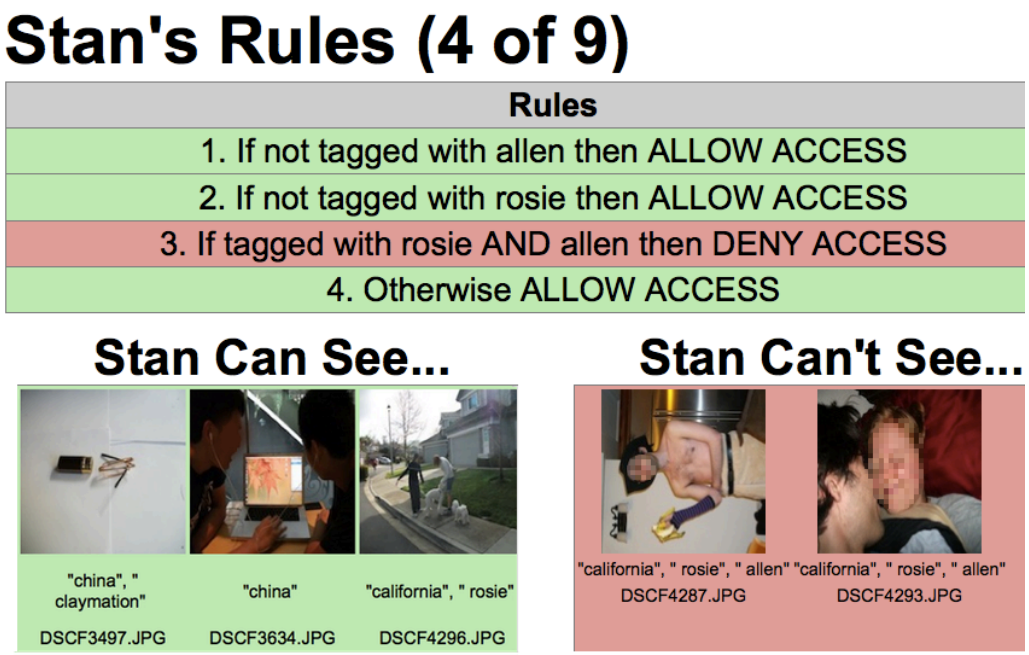

Figure 5.1: Interface for demonstrating machine-generated policy rules. The interface demonstrated both the machine-generated rules and the resulting effective access policy. (Names changed and faces blurred.)

photos. The participant's boss, by contrast, might be assigned a more complicated ruleset with three rules: "If tagged with *landscape*, then *allow*"; "if tagged with *work*, then *allow*"; and the default rule "otherwise deny."

We applied the machine-generated rulesets to the participant's photos, resulting in a set of allowed and denied photos for each friend. In the interest of time, we selected two example rulesets to present, including at least one example with at least one non-default rule. If no non-default rulesets were created, we showed the participant a default-only ruleset, as well as a generic non-default example prepared in advance.

Our goal with these examples was to familiarize the participant with the idea of tag-based access-control rules and get preliminary reactions, rather than to evaluate the success of these rules in detail. We designed a simple rule-display interface, shown in Figure 5.1, intended only to demonstrate the machine-generated rules and stimulate discussion. The interface displayed the text rules for a given friend at the top of the screen and a thumbnail photo display at the bottom. The photo display distinguished the photos the friend was and was not allowed to see under the rules. Each photo was displayed with its tags to help the participant understand how the rules were applied. This interface was not intended to simulate any real system for managing tag-based access control.

T2: Tagging for access control. After using the sample machine-generated rules to introduce the concept of tag-based access control, we returned to Picasa for T2. We invited the participant to add to and/or delete from the tags she had added in T1, with the joint objectives of finding photos more easily and creating tag-based access-control rules. As before, we observed the participant's tagging behavior and strategies using a think-aloud mechanism.

Detailed review of machine-generated rules. Next, we explored how successful the participant's T2 strategy had been and gathered detailed feedback on the tag-based access-control concept. We used the tags from T2 and the participant's access-control preferences to create a new set of machine-generated access-control rules for each friend. We showed the participant all the resulting rulesets, using the same interface, one friend at a time. For each friend, we asked detailed questions about how accurately the rules reflected the participant's preferences, including for other photos she had taken in the past or might take in the future.

We also used our interface's *"show conflicts" view* to highlight any photos that were misclassified. We asked how upset the participant was about the misclassifications and how they affected her view of the ruleset's overall success. We also asked how she might change the tags or the ruleset to more accurately reflect her preferences.

T3: Refinement and wrap-up. To examine what the participant had learned, we invited her to add to and/or delete tags from T2, this time keeping in mind what she had learned during the detailed rule review. Once again, the goal of the tagging was to make both finding photos and developing access-control rules easier. As before, a think-aloud mechanism helped us observe the participant's behavior and strategy. Although we created rules from these tags for post-processing and analysis, in the interest of time we did not show these rules to the participant. Finally, we asked each participant a series of general questions about her photo-tagging and sharing habits.

Machine-generated rules and analysis. We used machine-generated rules both to demonstrate tag-based access-control rules to our participants and to conduct post-interview analysis. For demonstration, we were mainly interested in creating rules that were somewhat human-readable and would provoke discussion (Q3). In post-interview analysis, we used the machine-generated rules to investigate Q1 and Q2: Could we construct reasonably well-fitting access-control rules from organizational tags, and would tags modified with access control in mind produce better rules? For this purpose, a rough approximation seemed sufficient, so we did not attempt to find an optimal rule-generating algorithm or

construct the best possible rules for each participant.

To achieve both goals, we created rules using an open-source implementation of the c4.5 decision-tree algorithm [97].[2] We trained the algorithm for each friend, lumping together weak and strong preferences into *allow* and *deny* categories and using default sensitivity settings. Photos with neutral preferences were ignored during training.

We displayed the results of the training to the participants and used them in our later analyses. We did not use separate training and test sets, because we wanted to establish a baseline scenario for how tags aligned with access-control policies. Future work might separately consider finding the optimal algorithm for generating rules, as well as how well rules generated from one set of photos could predict access-control policies for other photos.

We report the results of generating rules from the participants' original tags, as well as their tags from tasks T1, T2 and T3, in the Results and Analysis section.

5.2 Demographics, policies, and tags

Table 5.1 lists demographic information for our 18 participants. Half were men, and half were women. The subjects trended young (between 18-32) and technologically focused: 10 of the 18 self-identified as science, technology, engineering and math (STEM) professionals or students. This bias toward youth and STEM professions may limit the generalizability of this study. Other work related to tagging and access control has also focused on similarly young populations [125, 137].

Our 18 participants provided between 40 and 48 photos each and listed 7 to 10 friends, plus Public. Overall, they expressed 6847 access preferences, each for one combination of a photo and a friend. 15.7% of preferences were strong allow, 40.8% were weak allow, 11.0% were strong deny, 14.9% were weak deny, and the remaining 17.5% were neutral. The distribution of preferences, however, varied widely across participants. P11 was most permissive, allowing 87.8% of access combinations; P04 was most restrictive, denying 80.0% of access combinations.

In T1, participants used on average 2.6 total tags per photo; P07 used the most, with 5.0, while P08 used the fewest, with 1.0. It is also possible to count the number of unique tags (e.g., counting the tag "family" once whether it was used once or many times). We will refer to this count as "unique" tags. Considering only each participant's unique tags, the average was 1.2 per photo, with a minimum of 4 tags for 48 photos (P13) and a maximum of 130 tags for 40 photos (P03).

[2] http://www2.cs.uregina.ca/~dbd/cs831/notes/ml/dtrees/c4.5/tutorial.html

Code	Age	Gender	Occupation	Photos/year	Tag software
P01	23	F	STEM professional	1001-5000	Picasa
P02	20	F	engineering student	101-500	iPhoto
P03	27	F	service industry	501-1000	Picasa, Facebook, Tumblr
P04	32	F	STEM professional	1001-5000	Skydrive
P05	24	F	student	1001-5000	iPhoto
P06	23	M	engineering student	501-1000	Picasa
P07	22	M	engineering student	101-500	Picasa
P08	24	M	student	101-500	Picasa
P09	18	M	engineering student	5001+	Picasa
P10	24	M	STEM professional	5001+	Photoshop Album
P11	26	M	STEM professional	1001-5000	Flickr
P12	28	F	art, writing	5001+	Lightroom
P13	23	M	clothing designer	51-100	Twitter, Yfrog, Photobucket
P14	20	M	engineering student	1001-5000	Picasa
P15	19	F	music student	501-1000	Picasa
P16	29	F	anthropology student	1001-5000	iPhoto
P17	25	M	STEM professional	501-1000	Picasa
P18	18	F	art student	1001-5000	iPhoto

Table 5.1: Participant demographics for tag-based policy study.

5.3 Results and analysis

Our results show that tags created for organization can often be used to create reasonably effective access-control rules (Q1). Asking users to update their tagging schemes with access control in mind produced even more accurate rules, in many cases with only limited modifications to the tags (Q2). We also observed that most participants quickly understood tagging for access control and were able to develop and apply a modified tagging strategy (Q3).

In the first two subsections, we describe results related to Q1 and Q2, respectively. Taken together, qualitative results from both subsections also address Q3.

5.3.1 Organizational tags can express many access policies

Overall, organizational tags performed well as the basis for access-control policies. Rules generated automatically from the participants' T1 tags were highly accurate, resulting in few false allow or false deny conflicts and indicating that the tags were expressive enough to be useful for such policies. Overall, the rules generated conflicts for only 7.8% of non-neutral photo-friend combinations in T1, with a conflict rate under 5% for one-third of participants. The best case for organizational tags was no conflicts for P11, who had a very simple allow-most policy. The worst observed case was user P08, with 19% conflicts, due

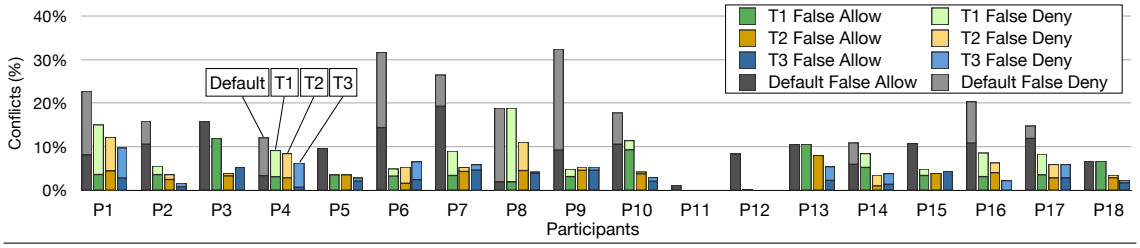

Figure 5.2: Conflict rate across tasks. Conflict rate is shown for all participants, distinguishing false allows and false denies. For each participant, we show four vertical bars, one each for the default policy, T1, T2, and T3. The default policy is the more accurate of allow all and deny all for each friend. For most participants, the conflict rate was highest for the default and decreased in consecutive tasks. P11 had no conflicts, and P12 had only 0.02% false deny conflicts in T2.

in part to his use of long, complex tags that were not repeated across photos. Figure 5.2 shows the rate of conflicts in each of the three tasks.

As a control, we compared our results to a simple default policy of either allow all or deny all, choosing the more accurate option for each participant-friend combination. In case of ties, we chose deny all to preserve privacy. This default policy, illustrated in Figure 5.2, produced more than twice as many conflicts as T1 (15.8% to 7.8%, significant, paired Wilcoxon, $\alpha = 0.05$).

Considering T1 conflicts in more detail, we found that for most conflicts (83.5%), the suggested rules disagreed with the less-serious weak preferences, bolstering the case that it is possible to make effective rules from organizational tags. To some extent, this reflects the fact that most non-neutral preferences were weak; however, we find that across participants, the proportion of conflicts with weak preferences was greater than the proportion of all preferences that were expressed as weak preferences (χ^2 per participant, aggregated using Fisher's combined test, $p < 0.05$).

In access control, false allows (erroneously granting access) are often of greater concern than false denies (erroneously denying access). In T1, 57% of conflicts were false allows. We might expect this to mirror the proportion of preferences that were deny preferences (since conflicts with those would be false allows); however, only 31% of preferences were deny preferences, a proportion significantly lower than that of the false allows (χ^2 and Fisher's combined test, as above, $p < 0.05$). More positively, most of those false allows conflicted only with weak preferences: just 13% of conflicts (less than 1% of decisions) were false allows that conflicted with a strong preference. We hypothesize that a classification algorithm tuned against false allows would ameliorate this further.

For most participants, a small number of photos were responsible for most conflicts. In T1, only 27% of all photos were misclassified by the machine-generated rules at least once, for any friend. The worst case was P04, for whom 60% of photos were misclassified at least

once; for more than two-thirds of participants, fewer than one-third of all photos were ever misclassified. This suggests that minor improvements in tagging or rule generation might reduce conflict rates considerably.

Simple rules from organizational tags. Another way to evaluate the performance of tag-based access-control policies is to consider their complexity; policies with too many rules or rules with too many clauses could prove incomprehensible to users. We measure rule complexity in two ways: by counting the number of non-default rules generated for each friend, and then by counting the number of unique tags used in those rules.

By both of these measures, the rules we generated were fairly simple. In T1, more than half of the 168 participant-friend combinations resulted in default-only rules. Among the 75 non-default cases, an average of 2.8 non-default rules were made for each friend. Even the worst case was relatively straightforward: the largest four rulesets contained only 4 non-default rules each. Examining the unique tags used in each ruleset yields similar results. On average, non-default rulesets for T1 contained only 2.4 unique tags. The worst case, rulesets containing six unique tags, occurred only twice among the 75 non-default rulesets.

It is also interesting to consider the relationship between accuracy and rule complexity. Prior to the study, we hypothesized that more-accurate rulesets would also be more complex, necessitating a tradeoff between expressiveness and ease of use. To examine this, we plot rule complexity (measured by unique tag count) against accuracy (measured by conflict rate), for all 168 participant-friend combinations, in Figure 5.3. The results show many instances of rulesets that are both simple and accurate: examples include allowing a spouse to see all photos, forbidding the public to see any photos, or allowing a boss to see only photos tagged with "work." We also see many rulesets in the top-left and bottom-right, indicating at least some tradeoff between accuracy and complexity, with few instances of rules that are both complex and inaccurate.

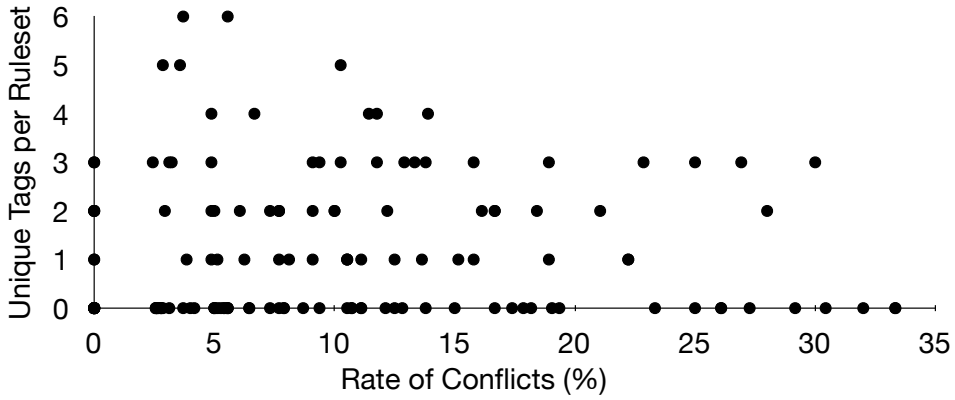

Figure 5.3: Rule complexity versus accuracy. Complexity (measured in unique tags per ruleset) versus accuracy (measured in conflict rate), for 168 participant-friend combinations. A value of zero unique tags denotes a default-only ruleset (deny all or allow all). In general, many rules are both simple and accurate, and few rules are both complex and inaccurate.

Reactions to sample rules. After creating machine-generated rules from the T1 tags, we showed each participant two rules and asked for general reactions. Many participants liked these rules and found them intuitive. For example, P05 said the rules for a former teacher, which denied access to photos from a friend's wedding and a graffiti-covered landmark, made sense because the rules "conceptualized what [she] was thinking." P17 said rules created for a friend with a child his son's age matched "the intuitive rule that [he] made" while setting up his preferences: the friend could see only photos containing P17's son.

However, the rules created from the organizational tags were not always fully successful. In some cases, the tags included in the rules were too general or too specific. For example, P09 said a rule denying his roommate access to photos tagged with "gf" was too general, as he wouldn't need to restrict all photos containing his girlfriend. Other participants were generally satisfied with the rules but flagged some exceptions. P14 said a rule for a former teacher "seem[ed] roughly accurate," but he was upset that the teacher could see one embarrassing, slightly lewd picture. In other cases, the machine-generated rules appeared coincidental, fitting the participant's preferences but using tags with little or no relation to the participant's policy decision-making.

Reactions to tag-based access control. After reviewing the sample rules, we asked participants for their overall impressions of tag-based access control, and found that the concept typically made sense. On a five-point Likert scale, 13 participants said the concept made complete sense or some sense (scores of 5 or 4), two were neutral (score of 3), and one said

the concept did not make sense (score of 2). Two others said it depended on circumstances (no score).

Among those who said the concept made complete sense, several said making policy using tags would save time or be "more efficient" (P03). Tag-based rules worked particularly well for P11, who had a subtle preference for preventing his family from seeing certain combinations of people: "If they can avoid seeing me and my girlfriend together, I'd probably use it for that."

P06, the only participant to say tag-based rules did not make sense, explained that he would need "a large number of tags to make it easier to make rules." P09, who chose neutral, expressed related concerns about scalability: "The results are great, but if you added more photos these rules would break down."

Ad-hoc access control with organizational tags. As another indicator that organizational tags may be appropriate for access-control policies, we found that several participants already used photo tagging to help implement their intended policies in various ad-hoc ways. P17 tagged his photos based mainly on the events at which they were taken; he used these tags to help him sort out which photos should be shared with whom on PicasaWeb. In the organization task, P01 tagged photos of herself with her boyfriend to keep track of things she didn't want her family to see.

Similarities between organizational and original tags. As described in Methodology, we also requested participants' original tags—that is, tags they added to the photos prior to the study—to confirm that the tags created in T1 were not highly different from tags the participants normally create. We were able to collect original tags for half our participants.

Our results indicate the two sets of organizational tags were reasonably similar. The overall rate of conflicts for rules generated from the original tags is 10.6%, compared to 8.5% for rules from T1 tags for the same 9 participants (not significant, paired Wilcoxon test, $p > 0.2$). This includes an outlier in P06, whose original photos included captions that were markedly different from the keyword tags he used in T1, resulting in a conflict rate of 25.9% for his original tags compared to 4.9% for his T1 tags. Original-tag conflict rates for the other 8 participants for whom this data was available were within 3 percentage points of the T1 rates. This provides a rough indication that T1 aligns with natural tagging behaviors for the majority of participants.

5.3.2 Tagging for access control provides improvement

Although access-control policies generated from organizational tags performed reasonably well, we found that policies generated from dual organization/access-control tags

performed even better. The overall conflict rate improved significantly[3] from 7.8% in T1 to 5.2% in T2, and the worst-case rate improved from 18.9% (P08) to 12.2% (P01). Across participants, the average improvement was 2.7 percentage points.

We observed a smaller but still significant improvement between T2 and T3, as participants fine-tuned their tags after viewing rules. The overall conflict rate improved to 4.2%, and the worst case improved to 9.8% (P01). Individual participants' conflict rates dropped 1.1 percentage points on average.

The ratios of weak to strong conflicts and of false allows to false denies did not change significantly among T1, T2 and T3. Unsurprisingly, participants were more concerned with false allows than false denies. During the detailed rule review, we asked participants to report how upset they were about each conflict on a five-point Likert scale. Comparing median upsetness per participant, we find that false allows cause significantly more upset (paired Wilcoxon test, $p < 0.01$).

Limited modifications for access-control tagging. Overall, participants made few modifications to their T1 tags in T2, averaging 33 modifications each, or less than one modification per photo. On average, participants only added or deleted 4 unique tags. Most of these changes (97%) were additions; only three participants deleted any tags in T2. P18 made the most modifications (108 additions, 5 unique); P11, by contrast, made no modifications.

Participants made even fewer refinements in T3, averaging 19 additions or deletions each (less than 0.5 per photo). P15 made the most (59, 7 unique), and P02 made the fewest (3, 2 unique). The refinement step was more balanced between addition and deletion; 65% of modifications were additions, and 10 participants deleted at least one tag. Figure 5.4 shows each participant's overall and unique modifications in both tasks.

Rule complexity across tasks. The overall pattern of avoiding complex, inaccurate rules was maintained across T2 and T3. The distribution of the complexity-accuracy tradeoff appeared to shift slightly, with more complex-but-accurate rules and fewer simple-but-inaccurate rules. This is attributable in part to a drop in default-only rulesets, from 93 in T1 to 66 in T2 and 59 in T3, as policies become more precise. However, this change was not signficant.

Strategies for access-control tagging. Participants appeared to follow one of three tagging strategies in T2: leveraging content-based tags for access control, using tags specifically de-

[3]Unless otherwise noted, significance tests in this section use a Friedman repeated-measures test to establish differences among the tasks, then paired Wilcoxon tests (chosen a priori) to separately compare T1 to T2 and T2 to T3. $\alpha = 0.05$.

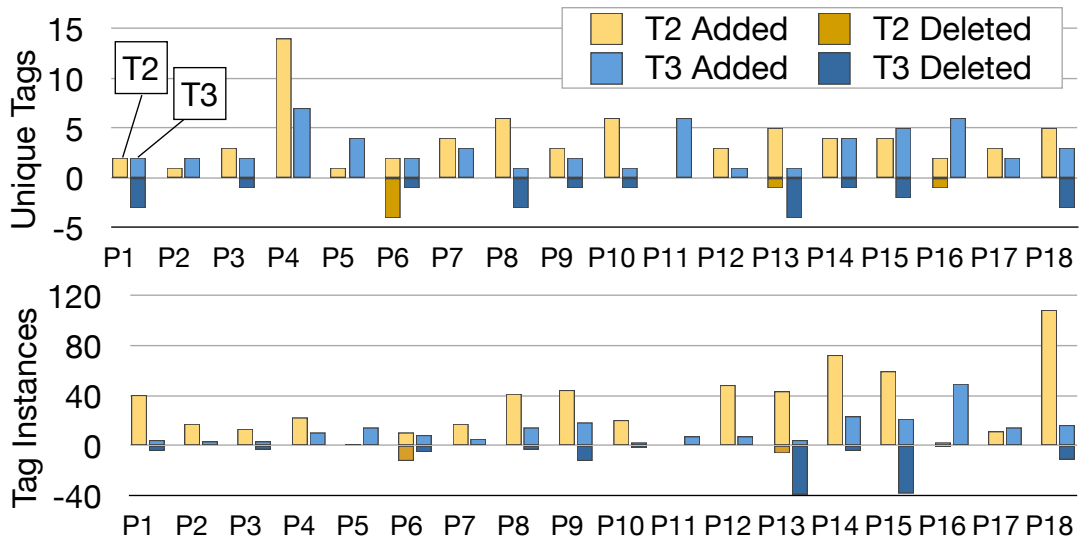

Figure 5.4: Tag additions and deletions. The unique tags and total tag instances added and deleted in T2 and T3, per participant.

signed to indicate access-control policy, or using a hybrid of content- and policy-based tags. Most participants articulated a clear strategy and applied it consistently, demonstrating that they had quickly grasped the concept of tag-based access control.

Five of 18 participants used the content-based strategy. At the extreme, P11 made no modifications because he felt his organizational tags were sufficient to specify his policy. P05 added "kiddos" to photos containing children, because their parents might not want those photos to be shared. P16 added "Belize" to one photo and changed "outside Sleeping Bear Lake" to just "Sleeping Bear Lake" on another; in both cases her modifications placed the photo in question into a group with other photos for which she had similar policy preferences. Participants using this strategy added, on average, 14 new tags and 4 new unique tags in T2.

Five other participants used an entirely policy-based strategy, creating tags indicating sharing policies like "private," "public," and "for friends." P09, who adopted this strategy, said he "would separate the two ideas [content and policy tags] completely." On average, participants using this strategy in T2 added 51 new tags and 4 new unique tags each.

The remaining eight participants used a hybrid strategy that combined policy- and content-based tags. Some of these participants used tags that conveyed both content and policy information, such as a "family" tag indicating family members could access photos containing family members. P07 used tags like these for family, close friends, and general friends. Others used both content and policy tags to form a combined policy: P10 wanted

to restrict photos with the "private" tag from most people, as well as photos with "strange" or "weird" content from some of his less-close friends. Participants using the hybrid strategy added, on average, 31 new tags and 4 new unique tags in T2.

During the detailed rule review after T2, we asked participants to suggest ways to improve the machine-generated rules. Many were able to articulate simple rules that matched their tagging strategies more closely than the automated rules. For example, P02 noted the machine-generated rules did not pick up on her strategy to restrict photos tagged "goofy" from less close friends.

Refinement strategies in T3. As previously noted, participants refining their tags after the detailed rule review made few additional modifications. The modifications they did make generally demonstrated a strong grasp of tag-based access control and a consistent approach to making tags that would facilitate better rules.

Most participants used T3 to adjust the granularity of their access-control-tagging scheme. For example, P03 originally tagged photos she didn't want made public with "drunk"; in T3, she added a "very drunk" tag to distinguish permissions for different levels of drunkenness. P18 changed policy tags on some photos to distinguish between "friends" and "close friends." Others used T3 to make rules more generalizable: P16 added "landscape" tags on top of "Sleeping Bear Lake" to make a broader category, "because if it's restricted to lake pictures," her friends were "not going to be seeing much."

Other participants added tags that were not necessary for creating policies for the photos and friends in the study, but could be useful for broader policies. For example, P17 added a "Pittsburgh" tag to photos taken in Pittsburgh; these photos were already classified correctly for the friends in the study, but he wanted to make a rule to share the photos explicitly with friends from Pittsburgh. T3 was also frequently used to make corrections or fix inconsistencies in tags. P10 consolidated "weird" and "strange" into one "weird" tag after noticing both were used in rules.

Only one participant overhauled her entire tagging scheme in T3: after viewing her rules, P15 switched from a policy-based strategy in which she assigned tags based on who *should* see the photos to an inverse strategy of tagging based on who *should not* see the photos.

5.4 Limitations

There are several important limitations to our study design. First, our results are limited by the participants we recruited and the photos they provided. As discussed in Demographics, our subject pool skewed young and technical; it also included only people willing to upload

their photos to our recruitment website. The generalizability of the photos provided was also limited by our request that participants upload previously tagged photos. We requested tagged photos so we could examine the ecological validity of tags created in our lab (as discussed in Results and Analysis). During recruitment, we attempted to encourage participants to upload a range of photos with different access preferences. We believe we succeeded, in part because participants expressed deny preferences for 25% of photo-friend combinations, including photos with sensitive content like alcohol, unprofessional behavior, and skinny dipping. We also asked participants about their current photo sharing preferences: 14 participants said they had published 80% or less of their study photos online. Most used various access-control mechanisms to protect photos they did publish, including setting privacy preferences on websites like Facebook and Picasa Web Albums. Participants distinguished clearly between these protected photos and photos published "publicly" on Tumblr, Twitter, personal websites and blogs. However, we acknowledge participants were unlikely to share their most private photos with us.

A second set of limitations concerns our use of machine-generated access-control rules. The algorithm has no access to the context and meaning of tags and no insight into the policy the participant intended when tagging for access control. As a result, some rules appeared strange or arbitrary to the participants, potentially driving them toward explicit policy-based tags like "private" and "public." We chose to use machine-generated rules to establish a standardized baseline for comparison accross tasks and provoke discussion with the participants. We also did not attempt to optimize our rule-generation mechanism or produce the best possible machine-generated rules. A better algorithm might result in fewer conflicts or strike a better balance among strong and weak conflicts and false allows and false denies.

Other limitations concern scale and generalizability. Our quantitative results measure only how well the rules fit the photos provided by the participants for the study; we talked to participants about how well the rules would generalize to other photos and friends, but cannot draw firm conclusions. Similarly, we cannot comment on whether the rules would remain tractable when dealing with thousands of photos and hundreds of friends, family members, and acquaintances.

5.5 Discussion

Our results indicate that tag-based access-control seems promising. In this section, we discuss potential design implications that arose from our findings and observations.

Automated rule generation. Participants were generally supportive of automated rule generation, and several explicitly said that they would like a system that suggested access-

control rules for their photos. As P12 put it, "That's kinda handy." Participants also did not seem to mind suggesting tweaks to automatically generated rules that did not completely capture their preferences. Although asking users to fill in an access-control grid for all their photo and preference combinations, as we did in the study, would be unrealistic in an actual system, such a system could potentially leverage users' willingness to tweak slightly incorrect rules by asking for a small set of ground-truth preferences over time and using these preferences to offer suggestions for baseline rules. Such a feature might be especially useful for people just starting to use a tag-based access-control system: results from our pilot testing indicate that users may better understand a tag-based rule system when it is demonstrated with their own tags and photos, rather than with generic examples.

Additionally, we did not try to optimize the machine-learning algorithm to reduce policy conflicts. Current work in privacy policy prediction promises to reduce conflicts further than observed in this work [125, 137].

Varied approaches to exception handling. Participants showed varying levels of preference flexibility when presented with tag-based rules. We categorize a user as *flexible* if, when presented with conflicts during the detailed rule review, she generally indicated this new access-control setting was also acceptable. In contrast, a *strict* user would attempt to modify her tags and propose new rules to resolve the majority of conflicts.

The 18 participants were evenly split between flexible and strict. P03 was representative of flexible users, saying, "If a couple more things got cut off than I intended, then it wouldn't matter to me so much." P10 felt similarly, explaining, "There's no reason for her to see relative photos, but I don't care too much." In contrast, P08 was representative of the strict users. When he encountered conflicts, he decided to create a number of additional tags and suggest associated rules to resolve those conflicts, saying, "One tag doesn't suffice for these three groups; [even] three tags are not enough."

A system design should account for people with both flexible and strict preferences. Stricter users could potentially be satisfied by providing an option for exception handling; however, providing such an option would need to be balanced with encouraging users to create generalizable rules to promote usability. P18 provides an example of this dynamic. She said that she had a large number of Facebook friends, and, while most of the time she would want to set permissions for groups, she would need some individual exceptions but not so many that the exceptions would became hard to manage.

Interface-supported rule management. Our results indicate that it is possible to repurpose organizational tags to create rules that capture the majority of a participant's preferences. However, a practical system must also help users create and manage rules and understand

the impact of tag and rule changes.

Although our rule-display interface was not intended to represent a real system, we asked participants for their impressions of it to gain some insight into potentially useful design features. As expected, most participants used the text rules to understand the future impact of the policy, and the photos to understand the immediate impact. P03 found both the rules and photos useful, saying he was worried about "not remembering how you tagged very specific photos ... I think seeing them is a more visceral response." A few participants mentioned that the photo display might be hard to use for larger sets of photos.

A tag-based access-control system could help users set policy by appropriately displaying relevant photos and rules, as well as demonstrating what would happen if the user changed tags or rules. The Expandable Grid [110], for example, could be used to demonstrate the impact of rule adjustments.

System support for manual tagging. Our users were able to actively and successfully engage in tagging for access control immediately after being exposed to tag-based access control (Q2). A practical system could further encourage tagging for access control in a variety of ways. One possibility is checking tags for consistency: for example, asking whether a user missed one tag in a group of photos she otherwise tagged similarly, noting slight changes in spelling, or highlighting use of close synonyms like P10's "strange" and "weird."

Additionally, a practical system could detect the types of tags frequently used in rules and remind the user to add these tags. We found that users tended to add descriptive, people, or permission tags when tagging for access control in T2. Displaying frequently used tags in such categories might nudge the user to create tags that are more useful for access-control rules. In addition, we investigated only user-provided keyword tags. However, person- and location-based tags were among the most common to appear in the access-control rules. Such tags are well supported by existing automation tools, including Picasa's face-detection feature (turned off during our study). Automated support for tagging is an emerging tool that could reduce user burden and help users add more, and more accurate, tags [122].

5.6 Summary

Overall, we found that tag-based rules are promising for use in an access-control system. Organizational tags can be repurposed to create reasonable access-control policies, and when participants actively create tags for access control, policies based on these tags are

yet more accurate. Participants are able to suggest and engage actively with tag-based rules.

These results suggest that it would be possible to create a usable access-control system with tag-based rules and minimal tagging overhead. It may be possible to additionally aid users with appropriate support for automated rule generation, exception handling, intuitive policy management, and automated tag generation and correction.

6 | A framework for usable access control

Our user studies, as well as previous work, identified several features that are important for meeting users' needs but largely missing in deployed access-control systems: for example, support for semantic policies, private metadata, and interactive policy creation (Chapter 5, [8, 109]).

In this chapter, I present Penumbra,[1] a distributed file system with access control designed to support users' policy needs while providing principled security. Penumbra provides for flexible policy specification meant to support real access-control policies, which are complex, frequently include exceptions, and change over time (Chapters 3 and 4, [15, 109, 130]). We define "usability" for Penumbra, which is not user-facing, as supporting specific policy needs and mental models that have been previously identified as important.

Penumbra's design is driven by three important factors. First, users often think of content in terms of its attributes, or *tags*—photos of my sister, budget spreadsheets, G-rated movies—rather than in traditional hierarchies (Chapter 5, [117, 120]). In Penumbra, both content and policy are organized using tags, rather than hierarchically.

Second, because tags are central to managing content, they must be treated accordingly. In Penumbra, tags are cryptographically signed first-class objects, specific to a single user's namespace. This allows different users to use different attribute values to describe and make policy about the same content. Most importantly, this design ensures tags used for policy specification are resistant to unauthorized changes and forgery. Policy for accessing tags is set independently of policy for files, allowing for private tags.

Third, Penumbra is designed to work in a distributed, decentralized, multi-user environment, in which users access files from various devices without a dedicated central server, an increasingly important environment [117]. We support multi-user devices; although these devices are becoming less common [34], they remain important, particularly in the home(Chapter 3, [72, 149]). Cloud environments are also inherently multi-user.

By combining semantic policy specification with logic-based credentials, Penumbra provides an intuitive, flexible policy model, without sacrificing correctness. Penumbra's

[1] designed and developed jointly with colleagues Yuan Liang, Lujo Bauer, Gregory R. Ganger, Nitin Gupta, and Michael K. Reiter

design supports distributed file access, private tags, tag disagreement between users, decentralized policy enforcement, and unforgeable audit records that describe who accessed what content and why that access was allowed. Penumbra's logic can express a variety of flexible policies that map well to real users' needs.

In this chapter, I describe Penumbra's architecture as well as important design choices.

6.1 High-level architecture

Penumbra encompasses an ensemble of devices, each storing files and tags. Users on one device can remotely access files and tags on other devices, subject to access control. Files are managed using semantic (i.e., tag-based) object naming and search, rather than a directory hierarchy. Users query local and remote files using tags, e.g., *type=movie* or *keyword=budget*. Access-control policy is also specified semantically, e.g., Alice might allow Bob to access files with the tags *type=photo* and *album=Hawaii*. Our concept of devices can be extended to the cloud environment. A cloud service can be thought of as a large multi-user device, or each cloud user as being assigned her own logical "device." Each user runs a *software agent*, associated with both her global public-key identity and her local uid, on every device she uses. Among other tasks, the agent stores all the *authorization credentials*, or cryptographically signed statements made by principals, that the user has received.

Each device in the ensemble uses a file-system-level *reference monitor* to control access to files and tags. When a system call related to accessing files or tags is received, the monitor generates a challenge, which is formatted as a logical statement that can be proved true only if the request is allowed by policy. To gain access, the requesting user's agent must provide a logical proof of the challenge. The reference monitor will verify the proof before allowing access. To make a proof, the agent assembles a set of relevant authorization credentials. The credentials, which are verifiable and unforgeable, are specified as formulas in an access-control logic, and the proof is a derivation demonstrating that the credentials are sufficient to allow access. Penumbra uses an intuitionistic first-order logic with predicates and quantification over base types, described further in Chapter 7.

The challenges generated by the reference monitors have seven types, which fall into three categories: authority to read, write, or delete an existing file; authority to read or delete an existing tag; and authority to create content (files or tags) on the target device. The rationale for this is explained in Section 6.2.1. Each challenge includes a nonce to prevent replay attacks; for simplicity, we omit the nonces in examples. The logic is not exposed directly to users, but abstracted by an interface that is beyond the scope of this thesis.

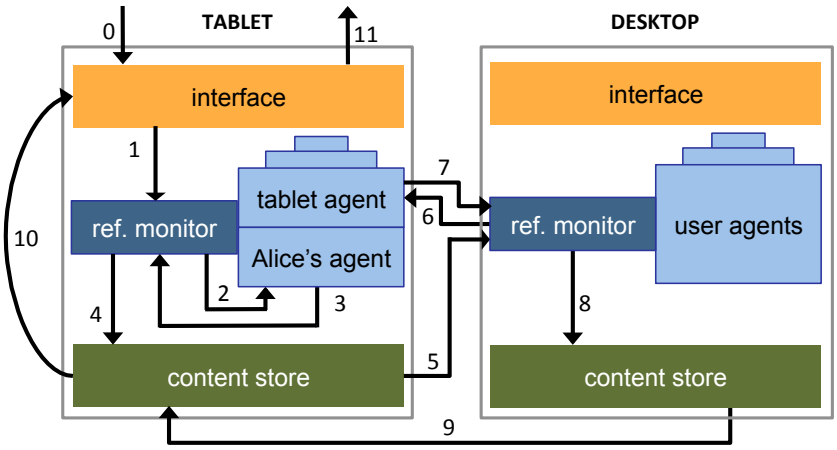

Figure 6.1: Access-control example. (0) Using her tablet, Alice requests to open a file stored on the desktop. (1) The interface component forwards this request to the reference monitor. (2) The local monitor produces a challenge, which (3) is proved by Alice's local agent, then (4) asks the content store for the file. (5) The content store requests the file from the desktop, (6) triggering a challenge from the desktop's reference monitor. (7) Once the tablet's agent proves the tablet is authorized to receive the file, (8) the desktop's monitor instructs the desktop's content store to send it to the tablet. (9–11) The tablet's content store returns the file to Alice via the interface component.

For both local and remote requests, the user must prove to her local device that she is authorized to access the content. If the content is remote, the local device (acting as client) must additionally prove to the remote device that the local device is trusted to store the content and enforce policy about it. This ensures that users of untrusted devices cannot circumvent policy for remote data. Figure 6.1 illustrates a remote access.

6.2 Design details

This section describes key design decisions related to handling tags, managing authority in a loosely-connected distributed system, enabling negative policies, and other topics.

6.2.1 Tags

Semantic management of access-control policy, in addition to file organization, gives new importance to tag handling. Because we base policy on tags, they must not be forged or altered without authorization. If Alice gives Malcolm access to photos from her Hawaiian vacation, he can gain unauthorized access to her budget if he can change its type from *spreadsheet* to *photo* and add the tag *album=Hawaii*. We also want to allow users to keep tags private and to disagree about tags for a shared file.

To support private tags, we treat each tag as an object independent of the file it describes.

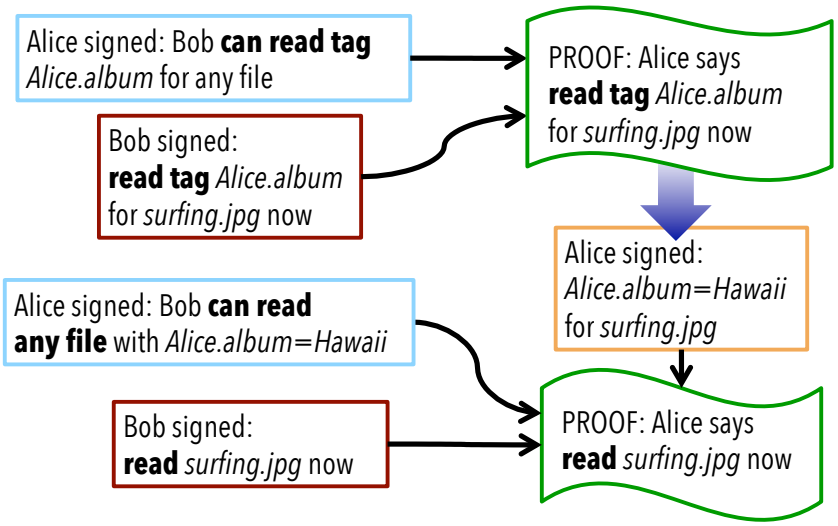

Figure 6.2: Example two-stage proof of access, expressed informally. In the first stage, Bob's agent asks which album Alice has placed the photo Luau.jpg in. After making the proof, Bob's agent receives a metadata credential saying the photo is in the album *Hawaii*. By combining this credential with Bob's authority to read some files, Bob's agent can make a proof that will allow Bob to open Luau.jpg.

Reading a tag requires a proof of access, meaning that assembling a file-access proof that depends on tags will often require first assembling proofs of access to those tags (Figure 6.2).

For tag integrity and to allow users to disagree about tags, we implement tags as cryptographically signed credentials of the form *principal* signed tag(*attribute*, *value*, *file*). For clarity in examples, we use descriptive file names; in reality, Penumbra uses globally unique IDs. For example, Alice can assign the song "Thriller" a four-star rating by signing a credential: Alice signed tag(rating, 4, "Thriller"). Alice, Bob, and Caren can each assign different ratings to "Thriller." Policy specification takes this into account: if Alice grants Bob permission to listen to songs where Alice's rating is three stars or higher, Bob's rating is irrelevant. Because tags are signed, any principal is free to make any tag about any file. Principals can be restricted from storing tags on devices they do not own, but if Alice is allowed to create or store tags on a device then those tags may reference any file.

Some tags are naturally written as attribute-value pairs (e.g., *type=movie*, *rating=PG*). Others are commonly value-only (e.g., photos tagged with *vacation* or with people's names). We handle all tags as name-value pairs; value-only tags are transformed into name-value pairs, e.g., from "vacation" to *vacation=true*.

Because tags are cryptographically signed, they cannot be updated; instead, the old credential is revoked (Section 6.2.4) and a new one is issued. As a result, there is no explicit write-tag authority.

6.2.2 Devices, principals, and authority

We treat both users and devices as principals who can create policy and exercise authority granted to them. Each principal has a public-private key pair, which is consistent across devices. This approach allows multi-user devices and decisions based on the combined trustworthiness of a user and a device. (Secure initial distribution of a user's private key to her various devices is outside the scope of this thesis.)

In Penumbra, the challenge statements issued by a reference monitor are of the form *device* says *action*, where *action* describes the access being attempted.[2] For Alice to read a file on her laptop, her software agent must prove that *AliceLaptop* says readfile(f).

This design captures the intuition that a device storing some data ultimately controls who can access it: sensitive content should not be given to untrusted devices, and trusted devices are tasked with enforcing access-control policy. For most single-user devices, a default policy in which the device delegates all of its authority to its owner is appropriate. For shared devices or other less common situations, a more complex device policy that gives no user full control may be necessary.

6.2.3 Negative policies

Negative policies, which forbid access rather than allow it, are important but often challenging for access-control systems. Without negative policies, many intuitively desirable rules are difficult to express. Examples taken from user studies include denying access to photos tagged with *weird* or *strange* (Chapter 5) and sharing all files other than financial documents(Chapter 3).

The first policy could naively be formulated as forbidding access to files tagged with *weird=true*; or as allowing access when the tag *weird=true* is not present. In our system, however, policies and tags are created by many principals, and there is no definitive list of all credentials. In such contexts, the inability to find a policy or tag credential does not guarantee that no such credential exists; it could simply be located somewhere else on the network. In addition, policies of this form could allow users to make unauthorized accesses by interrupting the transmission of credentials. Hence, we explore alternative ways of expressing deny policies.

Our solution has two parts. First, we allow delegation based on tag inequality: for example, to protect financial documents, Alice can allow Bob to read any file with *topic≠financial*. This allows Bob to read a file if his agent can find a tag, signed by Alice, placing that file

[2]The logical primitive says describes a belief held by a principal, as demonstrated either by her own cryptographic assertion of that belief, or by deriving that belief from other known cryptographic assertions. See Section 7.1 for further details.

into a topic other than financial. If no credential is found, access is still denied, which prevents unauthorized access via credential hiding. This approach works best for tags with non-overlapping values—e.g., restricting children to movies not rated R. If, however, a file is tagged with both *topic=financial* and *topic=vacation*, then this approach would still allow Bob to access the file.

To handle situations with overlapping and less-well-defined values, e.g., denying access to weird photos, Alice can grant Bob authority to view files with *type=photo* and *weird=false*. In this approach, every non-weird photo must be given the tag *weird=false*. This suggests two potential difficulties. First, we cannot ask the user to keep track of these negative tags; instead, we assume the user's policymaking interface will automatically add them (e.g., adding *weird=false* to any photo the user has not marked with *weird=true*). As we already assume the interface tracks tags to help the user maintain consistent labels and avoid typos, this is not an onerous requirement. Second, granting the ability to view files with *weird=false* implicitly leaks the potentially private information that some photos are tagged *weird=true*. We assume the policymaking interface can obfuscate such negative tags (e.g., by using a hash value to obscure *weird*), and maintain a translation to the user's original tags for purposes of updating and reviewing policy and tags. We discuss the performance impact of adding tags related to the negative policy (e.g., *weird=false*) in Chapter 9.

6.2.4 Expiration and revocation

In Penumbra, as in similar systems, the lifetime of policy is determined by the lifetimes of the credentials that encode that policy. To support dynamic policies and allow policy changes to propagate quickly, we have two fairly standard implementation choices.

One option is short credential lifetimes: the user's agent can be set to automatically renew each short-lived policy credential until directed otherwise. Alternatively, we can require all credentials used in a proof to be online countersigned, confirming validity [80]. Revocation is then accomplished by informing the countersigning authority. Both of these options can be expressed in our logic; we do not discuss them further.

6.2.5 Threat model

Penumbra is designed to prevent unauthorized access to files and tags. To prevent spoofed or forged proofs, we use nonces to prevent replay attacks and rely on standard cryptographic assumptions that signatures cannot be forged unless keys are leaked. We also rely on standard network security techniques to protect content from observation during transit between devices.

Penumbra employs a language for capturing and reasoning about trust assertions. If

trust is misplaced, violations of intended policy may occur—for example, an authorized user sending a copy of a file to an unauthorized user. In contrast to other systems, Penumbra's flexibility allows users to encode limited trust precisely, minimizing vulnerability to devices or users who prove untrustworthy; for example, different devices belonging to the same owner can be trusted differently.

7 | A logic for tag-based policy specification

The heart of Penumbra's access-control-enforcement mechanism is the logical policy language used to express policies and present proofs of authority. Penumbra's logic is developed as an extension of the Grey logic [21], with special predicates that support tag-based policy as well as standard file-system operations.[1]

In this chapter, I first introduce background on the basic concepts used across many access-control logics, including Penumbra's. In the second subsection, I present the details of Penumbra's logic, including its basic inference rules, the logical constructs used to express policy for files and tags, and a set of basic default policies that can be used to bootstrap a simple policy for single-owner devices.

7.1 Background: Using logic for access control

Access-control logics commonly use A signed F to describe a principal cryptographically asserting a statement F.

A says F describes beliefs or assertions F that can be derived from other statements that A has signed. In the simplest case, every signed statement can be used to directly derive a corresponding says statement:

$$\frac{A \text{ signed } F}{A \text{ says } F}$$

For simplicity, I generally elide this step throughout the rest of the thesis.

Additional conclusions about what A believes (says) can be derived using modus ponens, as follows:

$$\frac{A \text{ says } F \quad A \text{ says } (F \rightarrow G)}{A \text{ says } G}$$

Statements that principals can make include both delegation and use of authority. In the following example, principal A grants authority over some action F to principal B, and B wants to perform action F.

[1] Joint work with Yuan Liang, Lujo Bauer, Gregory R. Ganger, and Michael K. Reiter.

$$A \text{ signed deleg } (B, F) \tag{7.1}$$
$$B \text{ signed } F \tag{7.2}$$

These statements can be combined, as a special case of modus ponens, to prove that B's action is supported by A's authority:

$$\frac{(7.1) \quad (7.2)}{A \text{ says } F}\ delegE$$

7.2 The Penumbra logic

For Penumbra, we define a first-order, intuitionistic access-control logic, with predicates and quantification over base types. Our logic includes standard operators such as and, or, and implication, as well as some constructions commonly used in access control, such as delegation and defining groups of users. In addition, there are specialized predicates for describing file-system actions. In our logic, tags are considered first-class objects, so actions on files and actions on tags are treated separately.

7.2.1 Basic inference rules

The basic inference rules of our logic are as follows:

$$\frac{A \to B \quad A}{B}\ impE \qquad \frac{A \quad B}{A \wedge B}\ andR \qquad \frac{A \wedge B}{A}\ andE1 \qquad \frac{A \wedge B}{B}\ andE2$$

$$\frac{K \text{ signed } A}{K \text{ says } A}\ signedE \qquad \frac{A}{K \text{ says } A}\ saysR \qquad \frac{K \text{ says } A \quad A \to B}{K \text{ says } B}\ saysE$$

$$\frac{K \text{ deleg}(K.group, F) \quad K \text{ says } M \text{ speaksfor } K.group}{K \text{ deleg}(M, F)}\ speaksforE$$

The first four are straightforward. *signedE* is explained above. *saysR* asserts that any principal agrees to any statement that is true, while *saysE* allows implication to distribute over says. *speaksforE* allows a principal who has been assigned to a group to use that group's authority.

deleg, which as described above allows principals to delegate authority, is defined in terms of implication as follows, where *A* and *B* are principals and *N* is a nonce:

$$A \text{ deleg}(B, F) \equiv \forall N\text{: } B \text{ says goal}(F, N) \to A \text{ says goal}(F, N)$$

Given the above definition, the *delegE* rule shown in Section 7.1 can straightforwardly

be proved as a lemma. The nonce is used to prevent replay attacks; for simplicity, I elide it in further examples.

7.2.2 Creating tags and files

Unlike reading and writing, in which authority is determined per file or tag, authority to create files and tags is determined per device. Because files are organized by their attributes rather than in directories, creating one file on a target device is equivalent to creating any other. Similarly, a user with authority to create tags can always create any tag in her own namespace, and no tags in any other namespace. So, only authority to create any tags on the target device is required.

Attempts to create files or metadata incur challenges of the form *device* says *action*(*d*), where *d* is a device and *action* can be either createfile or createtag.

7.2.3 Semantic policy for files

There are four basic operations that can be applied to a file: creation, deletion, reading, and writing. As described in the previous subsection, file creation is managed on a per-device rather than a per-file basis. The other three file-access types incur challenges of the form *device* says *action*(*f*), where *f* is a file and *action* can be one of readfile, writefile, or deletefile.

A policy by which Alice allows Bob to listen to any of her music is implemented as a conditional delegation: If Alice says a file has *type=music*, then Alice delegates to Bob authority to read that file. We write this as follows:

$$\text{Alice signed } \forall f : \text{tag(type,music,}f\text{)} \rightarrow \text{deleg(Bob,readfile(}f\text{))} \tag{7.3}$$

To use this delegation to listen to "Thriller," Bob's agent must show that Alice says "Thriller" has *type=music*, and that Bob intends to open "Thriller" for reading, as follows:

$$\text{Alice signed tag(type,music,"Thriller")} \tag{7.4}$$

$$\text{Bob signed readfile("Thriller")} \tag{7.5}$$

$$\dfrac{\dfrac{(7.3) \qquad (7.4)}{\text{Alice says deleg(Bob,readfile("Thriller"))}} \qquad (7.5)}{\text{Alice says readfile("Thriller")}}$$

In this example, we assume Alice's devices grant her access to all of her files; we elide proof steps showing that the device assents once Alice does. We similarly elide instantiation of the quantified variable.

We can easily extend such policies to multiple attributes or to groups of people. To allow the group "co-workers" to view her vacation photos, Alice would assign users to

the group (which is also a principal) by issuing credentials as follows:

$$\text{Alice signed speaksfor(Bob, Alice.co-workers)} \tag{7.6}$$

Then, Alice would delegate authority to the group rather than to individuals:

$$\text{Alice signed } \forall f : \text{tag(type,music,}f\text{)} \rightarrow \text{deleg(Alice.co-workers,readfile(}f\text{))} \tag{7.7}$$

To apply this delegation, Bob can use *speaksforE*.

7.2.4 Policy for tags

Penumbra supports private tags by requiring a proof of access before allowing a user or device to read a tag. Because tags are central to file and policy management, controlling access to them without impeding file system operations is critical.

Attribute lists. Most tag actions apply to more than one tag at a time—for example, reading all tags associated with a given file, or querying for all files that have a certain tag. To handle this, we define tag challenges to have the form *device* says *action*(*attribute list*,*file*), where *action* is either readtags or deletetags. We define an *attribute list* as a set of (principal,attribute name,attribute value) triples representing the tags for which access is requested. Each such triple is one *attribute*.

Attribute lists consist of a *head* (one attribute) and a *tail* (an attribute list), concatenated together: *a*::*L*. The empty list is represented by nil.

We define axioms for attribute lists using standard list operations, as follows:

$$\frac{}{L \equiv L\text{:nil}}\ cat1 \qquad \frac{L'' \equiv L : L'}{a :: L'' \equiv (a :: L) : L'}\ cat2 \qquad \frac{}{a \in a\text{::}L}\ inList1$$

$$\frac{a \in L}{a \in b\text{::}L}\ inList2 \qquad \frac{}{\text{nil} \in L}\ inList3 \qquad \frac{a \in L \quad L' \in L}{a :: L' \in L}\ inList4$$

$$\frac{}{L \equiv L}\ equiv1 \qquad \frac{L \in L' \quad L' \in L}{L \equiv L'}\ equiv2$$

These axioms allow reasoning about list concatenation, sublists, and list equivalence. These operations are important for the Penumbra logic, because permission to a access an attribute list should hold regardless of the list's ordering. Further implications of subsetting and equivalence for tag permissions are discussed below.

For convenience, we also define an ownership predicate: if all attributes in a list are signed by the same principal, then that principal owns the list. This allows the creation of, for example, policies that enable users to always read their own tags. To accomplish this, we define the following inference rules:

$$\frac{}{K \text{ owns nil}}\ ownR1 \qquad\qquad \frac{K \text{ owns } L}{K \text{ owns } (K,n,v)::L}\ ownR2$$

Tag policy for queries. Common accesses to tags fall into three categories. A *listing query* asks which files belong to a category defined by one or more attributes, e.g., list all Alice's files with *type=movie* and *genre=comedy*. An *attribute query* asks the value of an attribute for a specific file, e.g., the name of the album to which a photo belongs. This kind of query can be made directly by users or by their software agents as part of two-stage proofs (Figure 6.2). A *status query*, which requests all the system metadata for a given file—last modify time, file size, etc.—is a staple of nearly every file access in most file systems (e.g., the POSIX *stat* system call).

Because tag queries can apply to multiple values of one attribute or multiple files, we use the wildcard * to indicate all possible completions. The listing query example above, which is a search on multiple files, would be specified with the attribute list [(Alice,type,movie), (Alice,genre,comedy)] and the target file *. The attribute query example identifies a specific target file but not a specific attribute value, and could be written with the attribute list [(Alice,album,*)] and target file "Luau.jpg." A status query for the same file would contain an attribute list like [(AliceLaptop,*,*)].

Credentials for delegating and using authority in the listing query example can be written as:

$$\text{Alice signed } \forall f : \text{deleg(Bob,readtags([(Alice,type,movie),(Alice,genre,comedy)],}f\text{))} \tag{7.8}$$

$$\text{Bob signed readtags([(Alice,type,movie),(Alice,genre,comedy)],*)} \tag{7.9}$$

These credentials can be combined to prove Bob's authority to make this query.

Implications of tag policy. One subtlety inherent in tag-based delegation is that delegations are not separable. If Alice allows Bob to list her Hawaii photos (e.g., files with *type=photo* and *album=Hawaii*), that should not imply that he can list all her photos or non-photo files related to Hawaii. However, tag delegations should be additive: a user with authority to list all photos and authority to list all Hawaii files could manually compute the intersection of the results, so a request for Hawaii photos should be allowed. Penumbra supports this subtlety, not as an axiom, but as part of the default policies established for each principal in the system (Section 7.2.5).

Another interesting issue is limiting the scope of queries. Suppose Alice allows Bob to read the album name only when *album=Hawaii*, and Bob wants to know the album name for "photo127." If Bob queries the album name regardless of its value (attributelist[(Alice,album,*)]),

no proof can be made and the request will fail. If Bob limits his request to the attribute list [(Alice,album,Hawaii)], the proof succeeds. If "photo127" is not in the Hawaii album, Bob cannot learn which album it is in.

Users may sometimes make broader-than-authorized queries: Bob may try to list all of Alice's photos when he only has authority for Hawaii photos. Bob's agent will then be asked for a proof that cannot be constructed. A straightforward option is for the query to simply fail. A better outcome is for Bob to receive an abridged list containing only Hawaii photos. One way to achieve this is for Bob's agent to limit his initial request to something the agent can prove, based on available credentials—in this case, narrowing its scope from all photos to Hawaii photos. We defer implementing this to future work.

7.2.5 Standard default policies

We know from research, including the study discussed in Chapter 3, that policy preferences vary widely among users. Nonetheless, for the common case of a device with one primary owner, there are some baseline policies appropriate for many situations which help to make bootstrapping a complete Penumbra policy easier. These policies are not built into the logic as axioms, but rather require creation of a set of "default" credentials when a device is initialized.

The basic set of default policies includes:

- The device and the device owner each allow readtags delegations to be combined, so that users can request the intersection of two allowable queries.
- The device owner can create files on this device.
- The device owner can create tags on this device.
- Any user can make a status query about any file she created.
- Any user can read and delete any tag in her namespace.
- Any user can read and write any file she created.

To provide the device owner with complete control over all files on her devices, we can add the following set of policies:

- The device owner can make a status query about any file on the device.
- The device owner can read, write, and delete any file on the device.
- The device owner can read and delete any tag on the device.

In addition, we can create a group of "trusted devices" for any user and make default

policy about it. This group, which should only contain single-user devices belonging to that user, is given authority to inherit all the user's permissions for all files and tags. This allows the user to seamlessly create, apply, and use a coherent policy across all of her devices.

Finally, it is often the case that when a user delegates authority over some category of files (specified via tags), she wants to simultaneously delegate authority to read the tags that define that category. For example, if Alice allows Bob to view files tagged with *type=photo* and *album=Hawaii*, we expect that in most cases she would also want to allow Bob to list which files are Hawaii photos. Tools that help Penumbra users make policy should provide the option to create these matching tag policies automatically.

8 | Realistic policy examples

In Section 7, I discussed abstractly how policy needs can be translated into logic-based credentials. In this chapter, I demonstrate that our infrastructure can represent real user policies.

It is difficult to obtain accurate policy data from users for use in testing. Historical usage data is hard to acquire, particularly for new sharing scenarios and access-control capabilities that have not yet been widely adopted. In lab settings, particularly with respect to these new capabilities, users struggle to articulate policies that capture real-life needs across a range of scenarios. As a result, there are no applicable standard benchmarks for personal policy and file-sharing.

Prior work has often, instead, relied on researcher experience or intuition [105, 113, 128, 147]. Such an approach, however, has limited ability to capture the needs of non-expert users [92].

To address this, colleagues[1] and I developed the first set of access-control-policy case studies that draw from target users' needs and preferences. They are based on detailed results from in-situ and experience-sampling user studies described in Chapters 3 and 4 and were compiled to realistically represent diverse policy needs. These case studies, which could also be used to evaluate other systems in this domain, are an important contribution of this thesis.

We draw on the HCI concept of *persona* development. Personas are archetypes of system users, often created to guide system design. Knowledge of these personas' characteristics and behaviors informs tests to ensure an application is usable for a range of people. Specifying individuals with specific needs provides a face to types of users and focuses design and testing [152].

To make the case studies sufficiently concrete for testing, each includes a set of users and devices, as well as policy rules for at least one user. Each also includes a simulated trace of file and metadata actions; some actions loosely mimic real accesses, and others test specific properties of the access-control infrastructure. Creating this trace requires

[1] Manya Sleeper, Lujo Bauer, Gregory R. Ganger, Nitin Gupta, and Michael K. Reiter

specifying many variables, including policy and access patterns, the number of files of each type, specific tags (access-control or otherwise) for each file, and users in each user group. We determine these details based on user-study data, and, where necessary, on inferences informed by HCI literature and consumer market research (e.g., [2, 143]). In general, the access-control policies are well-grounded in user-study data, while the simulated traces are more speculative.

An access-control system should support ...	**Sources**	**Case study**
access-control policies on metadata	[8, 29]	All
policies for potentially overlapping groups of people, with varied granularity (e.g., family, subsets of friends, strangers, “known threats")	[8, 29, 66, 102, 109, 124]	All
policies for potentially overlapping groups of items, with varied granularity (e.g., health information, “red flag" items)	Ch. 3, [66, 102, 109]	All
photo policies based on photo location., people in photo	Ch. 5, [8, 29]	Jean, Susie
negative policies to restrict personal or embarrassing content	Ch. 5 [8, 29, 109]	Jean, Susie
policy inheritance for new and modified items	[8, 124]	All
hiding unshared content	Ch. 4, [109]	All
joint ownership of files	Ch. 3 and 4	Heather/Matt
updating policies and metadata	[8, 29, 124]	—

Table 8.1: Access control system needs from literature.

In line with persona development [152], the case studies are intended to include a range of policy needs, especially those most commonly expressed, but not to completely cover all possible use cases. To verify coverage, we collated policy needs discussed in the literature. Table 8.1 presents a high-level summary. The majority of these needs are at least partially represented in all of our case studies. Unrepresented is only the ability to update policies and metadata over time, which Penumbra supports but we did not include in our test cases. The diverse policies represented by the case studies can all be encoded in Penumbra; this provides evidence that our logic is expressive enough to meet users' needs.

Case study 1: Susie. This case (Figure 8.1), drawn from the study described in Chapter 5, captures a default-share mentality: Susie is happy to share most photos widely, with the exception of a few containing either highly personal content or pictures of children she works with. As a result, this study exercises several somewhat-complex negative policies. This study focuses exclusively on Susie's photos, which she accesses from several personal devices but which other users access only via simulated “cloud" storage. No users besides Susie have write access or the ability to create files and tags. Because the original study collected detailed information on photo tagging and policy preferences, both the tagging and the policy are highly accurate.

SUSIE

Individuals: Susie, mom
Groups: friends, acquaintances, older friends, public
Devices: laptop, phone, tablet, cloud
Tags per photo: 0-2 access-control, 1-5 other
Policies:
Friends can see all photos.
Mom can see all photos except mom-sensitive.
Acquaintances can see all photos except personal, very personal, or red flag.
Older friends can see all photos except red flag.
Public can see all photos except personal, very personal, red flag, or kids.

Figure 8.1: Details of the Susie case study.

Case study 2: Jean. This case study (Figure 8.2) is drawn from the same user study as Susie. Jean has a default-protect mentality; she only wants to share photos with people who are involved in them in some way. This includes allowing people who are tagged in photos to see those photos, as well as allowing people to see photos from events they attended, with some exceptions. Her policies include some explicit access-control tags—for example, restricting photos tagged *goofy*—as well as hybrid tags that reflect content as well as policy. As with the Susie case study, this one focuses exclusively on Jean's photos, which she accesses from personal devices and others access from a simulated "cloud." Jean's tagging scheme and policy preferences are complex; this case study includes several examples of the types of tags and policies she discussed, but is not comprehensive.

JEAN

Individuals: Jean, boyfriend, sister, Pat, supervisor, Dwight
Groups: volunteers, kids, acquaintances
Devices: phone, two cloud services
Tags per photo: 1-10, including mixed-use access control
Policies:
People can see photos they are in.
People can see photos from events they attended.
Kids can only see *kids* photos.
Dwight can see photos of his wife.
Supervisor can see *work* photos.
Volunteers can see *volunteering* photos.
Boyfriend can see *boyfriend*, *family reunion*, and *kids* photos.
Sister can see *sister's wedding* photos.
Acquaintances can see *beautiful* photos.
No one can see *goofy* photos.

Figure 8.2: Details of the Jean case study.

Case study 3: Heather and Matt. This case study (Figure 8.3) is drawn from the study described in Chapter 3. Heather and Matt are a couple with a young daughter; most of the family's digital resources are created and managed by Heather, but Matt has full access. Their daughter has access to the subset of content appropriate for her age. The couple exemplifies a default-protect mentality, offering only limited, identified content to friends, other family members, and co-workers. This case study includes a wider variety of content, including photos, financial documents, work documents, and entertainment media. The policy preferences reflect Heather and Matt's comments; the assignment of non-access-control-related tags is less well-grounded, as they were not explicitly discussed in the interview.

HEATHER AND MATT

Individuals: Heather, Matt, daughter
Groups: friends, relatives, co-workers, guests in the house
Devices: laptop, two phones, DVR, tablet
Tags per item: 1-3, including mixed-use access control
Policies:
Heather and Matt can see all files
Co-workers can see all photos and music
Friends and relatives can see all photos, TV shows, and music
Guests can see all TV shows and music
Daughter can see all photos, all music, *not_inappropriate* TV shows
Heather can update all files except TV shows
Matt can update TV shows

Figure 8.3: Details of the Heather and Matt case study.

Case study 4: Dana. This case study (Figure 8.4) is drawn from the same user study as Heather and Matt. Dana is a law student who lives with a roommate and has a strong default-protect mentality. She has confidential documents related to a law internship that must be protected. This case study includes documents related to work, school, household management, and personal topics like health, as well as photos, e-books, television shows, and music. The policy preferences closely reflect Dana's comments; the non-access-control tags are drawn from her rough descriptions of the content she owns.

Case study 5: Joanna. This case study (Figure 8.5) is drawn from the same study as the previous two. Joanna is a college student who lives with three roommates, including her boyfriend. Joanna has a mix of default-share and and default-protect policies covering documents on various topics as well as photos and music, and she makes important

DANA

Individuals: Dana, sister, mom, boyfriend, roommate, boss
Groups: colleagues, friends
Devices: laptop, phone, cloud service
Tags per item: 1-3, including mixed-use access control
Policies:
Boyfriend and sister can see all photos
Friends can see *favorite* photos
Boyfriend, sister, friends can see all music and TV shows
Roommate can read and write household documents
Boyfriend and mom can see health documents
Boss can read and write all work documents
Colleagues can read and write work documents per project

Figure 8.4: Details of the Dana case study.

distinctions between read and write permissions. As with Heather/Matt and Dana, the policy preferences closely match Joanna's expressed preferences.

JOANNA

Individuals: Joanna, boyfriend, boss, professor
Groups: friends, parents
Devices: laptop, camera, phone, printer, media server, music player
Tags per item: 1-3 access-control, 0-1 other
Policies:
Everyone can listen to music.
Boyfriend and friends can see all photos.
Parents can see all photos except drinking.
Boyfriend can read and write public documents.
Boyfriend can read school and study-abroad documents.
Parents can read public, school, and study-abroad documents.
Friends can read public and school documents.
Boss can read and write work documents.
Boss can read public and school documents.
Professor can read and write school documents from his class.
Professor can read public and school documents.

Figure 8.5: Details of the Joanna case study.

9 | Implementation and evaluation

In this chapter, I discuss our implementation of the Penumbra design and our evaluation of its performance.[1]

9.1 File system implementation

Penumbra is implemented in Java, on top of FUSE [1]. Users interact normally with the Linux file system; FUSE intercepts system calls related to file operations and redirects them to Penumbra. Instead of standard file paths, Penumbra expects semantic queries. For example, a command to list G-rated movies can be written 'ls "query:Alice.type=movie & Alice.rating=G".'

Figure 9.1 illustrates Penumbra's architecture. System calls are received from FUSE in the front-end interface, which also parses the semantic queries. The central controller invokes the reference monitor to create challenges and verify proofs, user agents to create proofs, and the file and (attribute) database managers to provide protected content. The controller uses the communications module to transfer challenges, proofs, and content between devices. We also implement a small, short-term authority cache in the controller. This allows users who have recently proved access to content to access that content again without submitting another proof. The size and expiration time of the cache can be adjusted to trade off proving time with faster response to policy updates.

The implementation is about 15,000 lines of Java and 1800 lines of C. The primary trusted computing base (TCB) includes the controller (1800 lines) and the reference monitor (2500 lines)—the controller guards access to content, invoking the reference monitor to create challenges and verify submitted proofs. The file manager (400 lines) must be trusted to return the correct content for each file and to provide access to files only through the controller. The database manager (1600 lines) similarly must be trusted to provide access to tags only through the controller and to return only the requested tags. The TCB also includes 145 lines of LF (logical framework) specification defining our logic.

[1] Joint work with William Melicher, Lujo Bauer, Gregory R. Ganger, Nitin Gupta, and Michael K. Reiter

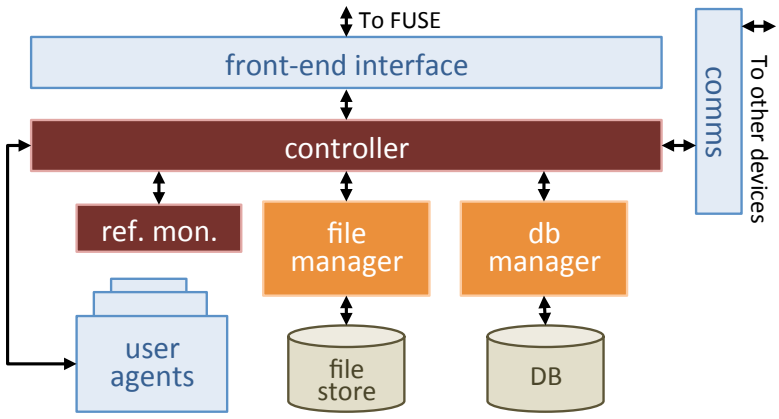

Figure 9.1: Penumbra architecture. The primary TCB (controller and reference monitor) is shown in red (darkest). The file and database managers (medium orange) also require some trust.

System call	Required proof(s)
mknod	create file, create metadata
open	read file, write file
truncate	write file
utime	write file
unlink	delete file
getattr	read tags: (system, *, *)
readdir	read tags: attribute list for *
getxattr	read tags: (principal, attribute, *)
setxattr	create tags
removexattr	delete tags: (principal, attribute, *)

Table 9.1: Proof requirements for file-related system calls.

Mapping system calls to proof goals. Table 9.1 shows the proof(s) required for each system call. For example, calling readdir is equivalent to a listing query—asking for all the files that have some attribute(s)—so it must incur the appropriate read-tags challenge.

Using "touch" to create a file triggers four system calls: getattr (the FUSE equivalent of stat), mknod, utime, and another getattr. Each getattr is a status query (see Section 7.2.4) and requires a proof of authority to read system tags. The mknod call, which creates the file and any initial metadata set by the user, requires proofs of authority to create files and metadata. Calling utime instructs the device to update its tags about the file. Updated system metadata is also a side effect of writing to a file, so we map utime to a write-file permission.

Disconnected operation. When a device is not connected to the Penumbra ensemble, its files are not available. Currently, policy updates are propagated immediately to all available devices; if a device is not available, it misses the new policy. While this is obviously impractical, it can be addressed by implementing eventual consistency (see for example Perspective [117] or Cimbiosys [108]) on top of the Penumbra architecture.

9.2 Proof generation and verification

Users' agents construct proofs using a recursive theorem prover loosely based on the one described by Elliott and Pfenning [47]. The prover starts from the goal (the challenge statement provided by the reference monitor) and works backward, searching through its store of credentials for one that either proves the goal directly or implies that if some additional goal(s) can be proven, the original goal will also be proven. The prover continues recursively solving these additional goals until either a solution is reached or a goal is found to be unprovable, in which case the prover backtracks and attempts to try again with another credential. When a proof is found, the prover returns it in a format that can be submitted to the reference monitor for checking. The reference monitor uses a standard LF checker implemented in Java.

The policy scenarios represented in our case studies generally result in a shallow but wide proof search: for any given proof, there are many irrelevant credentials, but only a few nested levels of additional goals. In enterprise or military contexts with strictly defined hierarchies of authority, in contrast, there may be a deeper but narrower structure. We implement some basic performance improvements for the shallow-but-wide environment, including limited indexing of credentials and simple fork-join parallelism, to allow several possible proofs to be pursued simultaneously. These simple approaches are sufficient to ensure that most proofs complete quickly; eliminating the long tail in proving time would require more sophisticated approaches, which we leave to future work.

User agents build proofs using the credentials of which they are aware. Our basic prototype pushes all delegation credentials to each user agent. (Tag credentials are guarded by the reference monitor and not automatically shared.) This is not ideal, as pushing unneeded credentials may expose sensitive information and increase proving time. However, if credentials are not distributed automatically, agents may need to ask for help from other users or devices to complete proofs (as in [21]); this could make data access slower or even impossible if devices with critical information are unreachable. Developing a strategy to distribute credentials while optimizing among these tradeoffs is left for future work.

9.3 Evaluation

To demonstrate that our design can work with reasonable efficiency, we evaluated Penumbra using the simulated traces we developed as part of the case studies from Chapter 8 as well as three microbenchmarks.

9.3.1 Experimental setup

We measured system call times in Penumbra using the simulated traces from our case studies. Table 9.2 lists features of the case studies we tested. We added users to each group, magnifying the small set of users discussed explicitly in the study interview by a factor of five. The set of files was selected as a weighted-random distribution among devices and access-control categories. For each case study, we ran a parallel control experiment with access control turned off—all access checks succeed immediately with no proving. These comparisons account for the overheads associated with FUSE, Java, and our database accesses—none of which we aggressively optimized—allowing us to focus on the overhead of access control. We ran each case study 10 times with and 10 times without access control.

During each automated run, each device in the case study was mounted on its own four-core (eight-thread) 3.4GHz Intel i7-4770 machine with 8GB of memory, running Ubuntu 12.04.3 LTS. The machines were connected on the same subnet via a wired Gigabit-Ethernet switch; 10 pings across each pair of machines had minimum, maximum, and median round-trip times of 0.16, 0.37, and 0.30 ms. Accounts for the people in the case study were created on each machine; these users then created the appropriate files and added a weighted-random selection of tags. Next, users listed and opened a weighted-random selection of files from those they were authorized to access. The weights are influenced by research on how the age of content affects access patterns [143]. Based on the file type, users read and wrote all or part of each file's content before closing it and choosing another to access. The specific access pattern is less important than broadly exercising the desired policy. Finally, each user attempted to access forbidden content to validate that the policy was set correctly and measure timing for failed accesses.

9.3.2 System call operations

Adding theorem proving to the critical path of file operations inevitably reduces performance. Usability researchers have found that delays of less than 100 ms are not noticeable to most users, who perceive times less than that as instantaneous [100]. User-visible operations consist of several combined system calls, so we target system call operation times well under the 100 ms limit.

Case study	Users	Files	Deleg. creds.	Proofs	System calls
Susie	60	2,349	68	46,646	212,333
Jean	65	2,500	93	30,755	264,924
Heather/Matt	60	3,098	101	39,732	266,501
Dana	60	3,798	89	27,859	74,593
Joanna	60	2,511	112	45,888	207,104

Table 9.2: Details of case study experiments. Proof and system call counts are averaged over 10 runs.

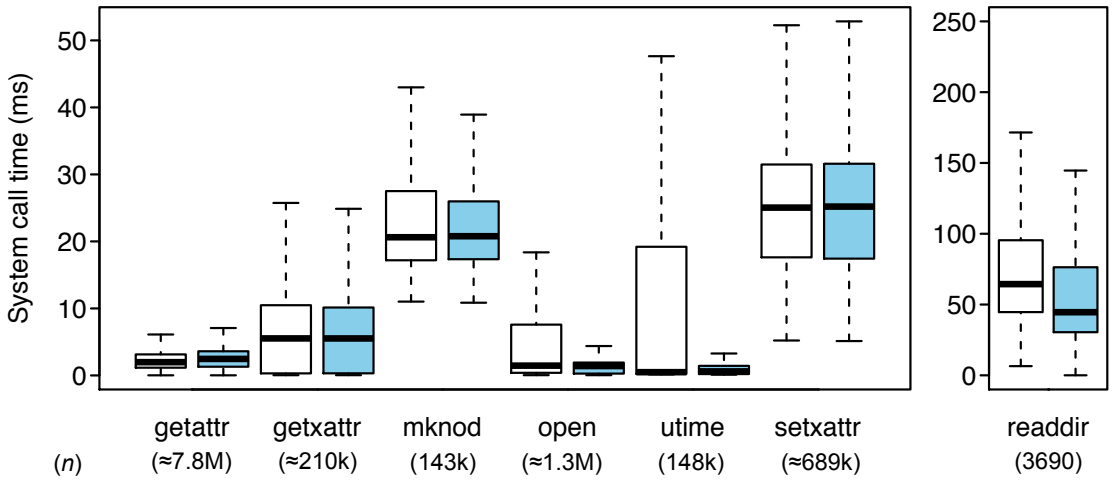

Figure 9.2: System call latencies. Shown with (white, left box of each pair) and without (shaded, right) access control, with the number of operations (n) in parentheses. ns vary up to 3% between runs with and without access control. Other than readdir (shown separately for scale), median system call times with access control are 1-25 ms and median overhead is less than 5%.

Figure 9.2 shows the duration distribution for each system call, aggregated across all runs of all case studies, both with and without access control. Most system calls were well under the 100 ms limit, with medians below 2 ms for getattr, open, and utime and below 6 ms for getxattr. Medians for mknod and setxattr were 21 ms and 25 ms. That getattr is fast is particularly important, as it is called within nearly every user operation. Unfortunately, readdir (shown on its own axis for scale) did not perform as well, with a median of 64 ms. This arises from a combination of factors: readdir performs the most proofs (one local, plus one per remote device); polls each remote device; and must sometimes retrieve thousands of attributes from our mostly unoptimized database on each device. In addition, repeated readdirs are sparse in our case studies and so receive little benefit from proof caching. The results also show that access-control overhead was low across all system calls. For open

and utime, the access control did not affect the median but did add more variance.

In general, we did little optimization on our simple prototype implementation; that most of our operations already fall well within the 100 ms limit is encouraging. In addition, while this performance is slower than for a typical local file system, longer delays (especially for remote operations like readdir) may be more acceptable for a distributed system targeting interactive data sharing.

9.3.3 Proof generation

Because proof generation is the main bottleneck inherent to our logic-based approach, it is critical to understand the factors that affect its performance. Generally system calls can incur up to four proofs (local and remote, for the proofs listed in Table 9.1). Most, however, incur fewer—locally opening a file for reading, for example, incurs one proof (or zero, if permission has already been cached). The exception is readdir, which can incur one local proof plus one proof for each device from which data is requested. However, if authority has already been cached no proof is required. (For these tests, authority cache entries expired after 10 minutes.)

Proving depth. Proving time is affected by proving depth, or the number of subgoals generated by the prover along one search path. Upon backtracking, proving depth decreases, then increases again as new paths are explored. Examples of steps that increase proving depth include using a delegation, identifying a member of a group, and solving the "if" clause of an implication. Although in corporate or military settings proofs can sometimes extend deeply through layers of authority, policies for personal data (as exhibited in the user studies we considered) usually do not include complex redelegation and are therefore generally shallow. In our case studies, the maximum proving depth (measured as the greatest depth reached during proof search, not the depth of the solution) was only 21; 9% of observed proofs (165,664 of 1,927,102) had depth greater than 10.

To examine the effects of proving depth, we developed a microbenchmark that tests increasingly long chains of delegation between users. We tested chains up to 60 levels deep. As shown in Figure 9.3a, proving time grew linearly with depth, but with a shallow slope—at 60 levels, proving time remained below 6 ms.

Red herrings. We define a *red herring* as an unsuccessful proving path in which the prover recursively pursues at least three subgoals before detecting failure and backtracking. To examine this, we developed a microbenchmark varying the number of red herrings; each red herring is exactly four levels deep. As shown in Figure 9.3b, proving time scaled approximately quadratically in this test: each additional red herring forces additional

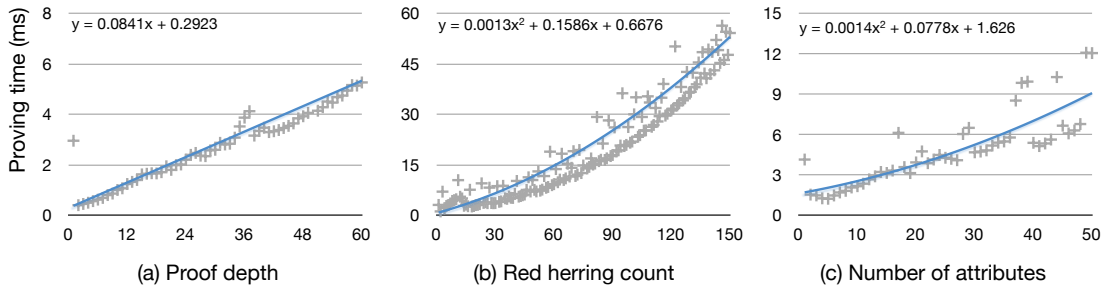

Figure 9.3: Proving time scalability. Three microbenchmarks showing how proving time scales with proving depth, red herrings, and attributes-per-policy. Shown with best-fit (a) line and (b,c) quadratic curve.

searches of the increasing credential space. In our case studies, the largest observed value was 44 red herrings; proofs with more than 20 red herrings made up only 2% of proofs (38,048 of 1,927,102). For up to 20 red herrings, proving time in the microbenchmark was generally less than 5 ms; at 40, it remained under 10 ms.

Proving time in the case studies. In the presence of real policies and metadata, changes in proving depth and red herrings can interact in complex ways that are not accounted for by the microbenchmarks. Figure 9.4 shows proving time aggregated in two ways. First, we compare case studies. Heather/Matt has the highest variance because files are jointly owned by the couple, adding an extra layer of indirection for many proofs. Joanna has more and more complex policy credentials than other case studies, leading to more red herrings and consequently longer proving times. Susie has a higher median and variance than Dana or Jean because of her negative policies, which also lead to more red herrings. Second, we compare proof generation times, aggregated across case studies, based on whether a proof was made by the primary user, by device agents as part of remote operations, or by other users. Most important for Penumbra is that proofs for primary users be fast, as users do not expect delays when accessing their own content; these proofs had a median time less than 0.52 ms in each case study. Also important is that device proofs are fast, as they are an extra layer of overhead on all remote operations. Device proofs had median times of 1.1-2.2 ms for each case study. Proofs for other users were slightly slower, but had medians of 2-9 ms in each case study.

We also measured the time it takes for the prover to conclude no proof can be made. Across all experiments, 1,566,309 instances of failed proofs had median and 90th-percentile times of 9 and 40 ms, respectively.

Finally, we consider the long tail of proving times. Across all 50 case study runs, the 90th-percentile proof time was 11 ms, the 99th was 44 ms, and the maximum was 1531 ms.

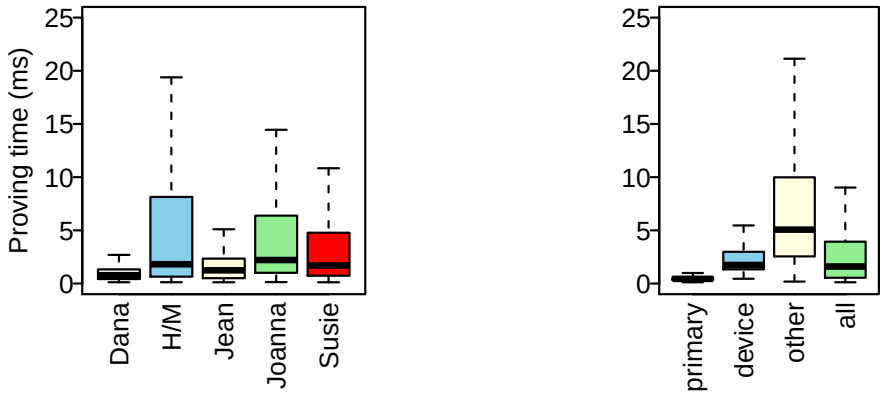

Figure 9.4: Details of proving times in the case studies. Organized by (left) case study and (right) primary user, device, and other users.

Of 1,908,800 proofs, 3,349 (0.2%) took longer than 100 ms. These pathological cases may have several causes: high depth, bad luck in red herrings, and even Java garbage collection. Reducing the tail of proving times is an important goal for future work.

Effects of negative policy. Implementing negative policy for attributes without well-defined values (such as the allow *weird=false* example from Section 6.2.3) requires adding inverse policy tags to many files. A policy with negative attributes needs $n \times m$ extra attribute credentials, where n is the number of negative attributes in the policy and m is the number of affected files.

Users with default-share mentalities who tend to specify policy in terms of exceptions are most affected. Susie, our default-share case study, has five such negative attributes: *personal*, *very personal*, *mom-sensitive*, *red-flag*, and *kids*. Three other case studies have one each: Jean restricts photos tagged *goofy*, Heather and Matt restrict media files tagged *inappropriate* from their young daughter, and Joanna restricts photos that depict her drinking from her parents. Dana, an unusually strong example of the default-protect attitude, has none. We also reviewed detailed policy data from [78] and found that for photos, the number of negative tags ranged from 0 to 7, with median 3 and mode 1. For most study participants, negative tags fall into a few categories: synonyms for private, synonyms for weird or funny, and references to alcohol. A few also identified one or two people who prefer not to have photos of them made public. Two of 18 participants used a wider range of less general negative tags.

The value of m is determined in part by the complexity of the user's policy: the set of files to which the negative attributes must be attached is the set of files with the positive

attributes in the same policy. For example, a policy on files with *type=photo & goofy=false* will have a larger m-value than a policy on files with *type=photo & party=true & goofy=false*.

Because attributes are indexed by file in the prover, the value of n has a much stronger affect on proving time than the value of m. Our negative-policy microbenchmark tests the prover's performance as the number of attributes per policy (and consequently per file) increases.

Figure 9.3c shows the results. Proving times grew approximately quadratically but with very low coefficients. For policies of up to 10 attributes (the range discussed above), proving time was less than 2.5 ms.

Adding users and devices. Penumbra was designed to support groups of users who share with each other regularly—household members, family, and close friends. Based on user studies, we estimate this is usually under 100 users. Our evaluation (Section 9.3) examined Penumbra's performance under these and somewhat more challenging circumstances. Adding more users and devices, however, raises some potential challenges.

When devices are added, readdir operations that must visit all devices will require more work; much of this work can be parallelized, so the latency of a readdir should grow sub-linearly in the number of devices. With more users and devices, more files are also expected, with correspondingly more total attributes. The latency of a readdir to an individual device is approximately linear in the number of attributes that are returned. Proving time should scale sub-linearly with increasing numbers of files, as attributes are indexed by file ID; increasing the number of attributes per file should scale linearly as the set of attributes for a given file is searched. Adding users can also be expected to add policy credentials. Using a microbenchmark (Figure 9.5), we demonstrate that users can be added to existing groups with sub-linear overhead; this is because the prover algorithm is specialized to recognize and sort through group membership credentials faster than most other types of credentials.

More complex policy additions can have varying effects. If a new policy is mostly disjoint from old policies, it can quickly be skipped during proof search, scaling sub-linearly. However, policies that heavily overlap may lead to increases in red herrings and proof depths; interactions between these could cause proving time to increase quadratically (see Figure 9.3) or faster. Addressing this problem could require techniques such as pre-computing proofs or subproofs [22], as well as more aggressive indexing and parallelization within proof search to help rule out red herrings sooner.

In general, users' agents must maintain knowledge of available credentials for use in proving. Because they are cryptographically signed, credentials can be up to about 2 kB in size. Currently, these credentials are stored in memory, indexed and preprocessed

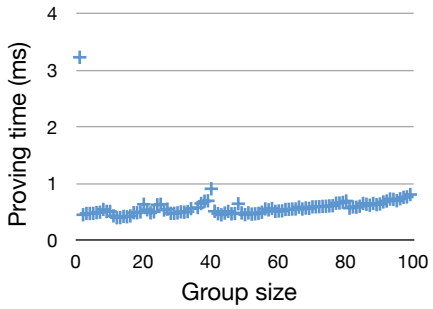

Figure 9.5: Group size scalability. A microbenchmark showing that adding users to existing groups has limited effect on proving time.

in several ways, to streamline the proving process. As a result, memory requirements grow linearly, but with a large constant, as credentials are added. To support an order of magnitude more credentials would require revisiting the data structures within the users' agents and carefully considering tradeoffs among insertion time, deletion time, credential matching during proof search, and memory use.

10 | Lessons learned: Requirements for access control for personal data

My work gathering user data and developing Penumbra has provided considerable insight into how access control for personal data typically fails and what will be required to achieve a more effective access-control environment. There are three main categories of failure: ad-hoc security that does not properly enforce desired policy; policy specification that cannot sufficiently express users' ideal policies; and policy management that requires too much effort from users.

Penumbra makes progress in the first two categories, but fully solving all three problems will require additional improvement to the design and implementation of access-control mechanisms, from the underlying policy-enforcement architecture all the way up to user-facing policy-management tools. In this chapter, I identify eight critical requirements for such improvements, spanning the three categories of failure identified above.[1]

Many existing prototypes and research systems (including Penumbra) implement some of these requirements, but no approach of which I am aware integrates all of them. In particular, many systems focus on the underlying security infrastructure, while others focus on the user interface; few combine the two. Designing a system that supports all these requirements will require bridging the security and human-computer-interaction communities to treat access-control holistically.

10.1 Requirements for security

When designing a new access-control system, the first concern must be what resources are protected and how that protection is enforced. We identify three requirements here: principled security, protection for metadata, and protection for policy.

10.1.1 Principled security

In access control, as with other areas of computer security, ensuring correct system behavior at all times is difficult. While formal methods are not a panacea, they can provide benefits

[1] Joint work with Lujo Bauer, Gregory R. Ganger, and Michael K. Reiter.

ranging from a more rigorous specification of what correct behavior should be to avoiding problems that might be overlooked in more ad-hoc approaches [144].

As discussed repeatedly in this thesis, access-control policies are becoming increasingly widespread and complex, suggesting that the gap between ad-hoc systems and their intended behavior will only grow. To combat this, new access-control systems should be built using formal methods that can provide provable correctness and completeness for at least some aspects of the system.

Perhaps the best-known formal approach to access control is logic-based access control, which is used in Penumbra and described in detail in Chapters 7 and 2. Among the many benefits to this approach, the requirement of a formal proof of access reduces the chance of a mismatch between the policy as specified and the behavior of the system implementation. (Of course, there is still the possibility of mismatch between policy as intended and as specified; we discuss ways to improve policymaking in Section 10.3.)

10.1.2 Protection for metadata

As people acquire ever-larger amounts of personal data, more metadata is needed to organize and manage it. Some of this metadata is user-defined—for example, rating songs, tagging photos, and manually tagging emails to sort them—while metadata like geolocation coordinates, facial recognition for photos, and document keyword assignment can be created automatically with consistently improving accuracy.

This metadata is often sensitive, and users may want to restrict access to it. For example, Facebook users often "untag" themselves from others' photos when they do not want to be connected with them [29]. Prior work has also found that users often consider the geolocation data attached to photos to be sensitive [8].

Access-control solutions, however, often consider metadata as part of an object, shared or restricted with the same policy as the content itself. We contend that this approach is wrong—metadata sensitivity may be independent of content sensitivity. For example, fugitive John McAfee had no problem with the content of a photo taken by a reporter, but the geolocation data attached to the photo gave away his location [141]. Beyond this sort of problem, knowledge about the ways we organize our content and the ratings or reviews we give to various items can reveal more information than the content itself. As a result, we believe it's critical for access-control systems to provide protection for metadata equivalent to but independent of protection for content.

The need to protect metadata has been partially considered via attribute-based encryption (e.g., [126]), in which sensitive metadata and personally-identifiable information are encrypted separately from the rest of a document's content. In Penumbra, we treat

metadata (tags) as first-class objects requiring separate proofs of access (Chapter 6). Both approaches have promise for managing this important aspect of access control for personal data.

It is important to note that providing independent access control for metadata creates even more policy for users to manage, a problem that we discuss in Section 10.3.

10.1.3 Protection for policy

Another subtle issue related to access control is the question of privacy for access-control policy itself. Often, policy statements—that is, rules that govern when access is allowed—themselves give away important information, especially in the realm of personal data where balancing social relationships can be key [145].

Some policy statements are sensitive because they imply information about content that is not made available; for example, allowing a friend access to all photos except nude photos suggests that nude photos exist. Relatedly, any negative policy (allowing access to all content except some subset) implies that the delegee is receiving only limited permissions; in our study of reactive policy creation (Chapter 4), we found that, unsurprisingly, there is often social awkwardness associated with denying access or providing only limited access.

In addition, policy about personal data often involves use of groups, which are frequently defined by tie strength [70, 77]. Classifying closer and less-close friends can be highly sensitive. Google+ takes this into account by not revealing group membership. Significant work has been done on restricting disclosure of access-control policy and credentials during trust negotiation (e.g., [96, 150]), but these approaches are rarely applied in the context of personal or home data. The KNOW model uses meta-policy to establish the sensitivity of policy rules and provide feedback to users about why their requests were denied [71].

10.2 Requirements for policy primitives

The next important consideration for new access-control systems is what kinds of policies should be supported. We think of this in terms of defining the abstractions that are available in the system to express users' desired policies. We distinguish the types of policy that a system can express from mechanisms for helping users properly use this available expressiveness. We identify two requirements: semantic organization and flexible policy primitives.

10.2.1 Semantic organization

Because users now have so much personal content, when making policy it becomes essential to organize or categorize that content rather than trying to make policy about each item individually. As discussed throughout this thesis, users often organize and share content based on its semantic attributes, rather than more traditional hierarchical naming schemes [117, 120]. In addition, hierarchical schemes fail when categories overlap.

In Section 2.3.2, I discussed prior work investigating the use of tag-based policies for access control. Our study of photo tagging for access control (Chapter 5) demonstrated that this approach has promise, and it makes up the core of Penumbra's design (Chapter 6).

10.2.2 Flexible policy primitives

Several studies, including our needs-assessment study (Chapter 3), have found that users' ideal policies are complex, nuanced, and have many exceptions [102, 124]. A system for controlling access to personal data must therefore support flexible, expressive policy specification. Semantic organization, as discussed above, provides one axis of flexibility, but not the only potentially important one.

Both our needs assessment and other studies [77] have identified who is present as a potentially important factor for determining access policy. Other possible primitives include where the access is taking place and what device is used to perform the access.

Access-control logics (such as [51]) that are designed to enable policies conditioned on system state could be straightforwardly extended to support location- and presence-based policies, assuming the availability of trustworthy sensors that can detect who is present or where the user is.

As we discuss in Chapter 4, policy is often dynamic and situational. Users who adjust policy for unusual situations may wish to grant single-use permissions; this mechanism can also be supported by logic-based access control [33].

10.3 Requirements for creating and updating policy

For any access-control system to be usable, it must be easy and low-effort for users to interact with it to accurately specify and change policy. In this section, we identify three requirements: dynamic and contextual policy creation, usable policy interfaces, and policy inference.

10.3.1 Dynamic and contextual policy creation

Prior work has shown that access-control policies are frequently dynamic, and that users often do not know what the correct policy should be until a particular situation arises.

Policies can change over the course of document lifecycles, due to changes in relationships, or due to transitions between identity management and personal archiving [109, 151]. Social network users often remove or restrict access to particular content that they regret posting [123, 139]. Participants in our needs-assessment study (Chapter 3)also showed strong interest in replicating social cues and behavioral norms from the physical world for digital content sharing.

These findings about dynamic and contextual policy suggest two important modes of policy creation: reactive and iterative. As discussed in Chapter 4, our experience-sampling study found that reactive policy creation shows promise for providing a stronger sense of control and supporting social cues, with limited annoyance. Wang et al. prototyped a system for implementing reactive policy creation for web content [138]. As demonstrated by the Grey system, reactive policy creation can be straightforwardly implemented using logic-based access control [21].

Because situations change, users must also be able to iteratively update their policies as needed. To this end, users often express interest in keeping audit data available, even if they rarely check it [77]. Logic-based access control can provide meaningful audit data that shows not just who accessed what, but also why the access was granted [135].

10.3.2 Usable policy interfaces

Supporting the dynamic and contextual policies discussed in the previous sections will require usable tools for viewing, interpreting, and updating policy preferences. Several researchers have suggested new approaches to policy visualization, including tools for configuring Facebook policy [30, 140] and expandable grids for allowing file-system policy to be quickly visualized and updated [110]. Optimal solutions for policy management would allow users to visualize policy at varying levels of detail, highlight the most important consequences of the current policy, and allow users to immediately and directly update any misconfigurations that are discovered.

Optimal solutions must also take into account the added complexity of managing policy for metadata and for policy statements themselves. For many users, these complications will only sometimes be useful; an optimal interface must keep from overwhelming the user with options. However, when these features are needed they are important, so interfaces to manage them must also be clearly understandable.

In addition, policymaking interfaces should cleanly present audit data for use in policy iteration, with emphasis on the most important or anomalous events. As before, this information should be kept out of the way when it isn't needed, but should be easy to access and understand when it is.

10.3.3 Policy inference

Configuring access-control policy is a secondary task users rarely want to spend time on. In addition, the overwhelming amount of personal digital content, combined with the large number of people who can potentially access content remotely and the wide variety of policies, can make setting up detailed policies a daunting task. To combat this, we will need techniques for suggesting and inferring policies on a user's behalf. Such inferences should not be completely autonomous, but should (at least sometimes) present policy decisions to users for review and modification.

Techniques for correctly inferring and recommending policy are not yet completely viable, but progress is being made. Researchers have prototyped tools for assigning a user's friends and acquaintances to access-control groups, both via fully automated techniques and by making intelligent suggestions for aiding manual classification [9, 95]. Fang and LeFevre developed a "privacy wizard" to infer social network privacy policies with minimal user input [48]. Significant effort (e.g., [148]) has been spent on inferring policy from logs, personnel data, and other soures. Other research (e.g., [23]) has demonstrated that access-control configuration errors can be detected automatically (and fixes suggested to the user) using machine learning techniques. User-adjustable learning techniques have been applied in the context of location-sharing policies [40, 75].

Other interesting directions to pursue related to policy inference might include policy recommendations derived from other, similar users or from experts' decisions, as well as inferring when it is most useful to alert to the user that her policies or audit data need review.

11 | Conclusion

As people create and share more and more personal digital content, effectively managing who can access that content under what circumstances becomes increasingly important for managing the presentation of one's identity across social and professional groups. At the same time, configuring access-control remains a secondary task users have little time or inclination to undertake, meaning that without intervention, the gap between users' ideal policies and their ability to realize them will only get larger. This thesis examines how users think about and may want to manage access to their digital content and presents Penumbra, a prototype distributed file system that provides many of features and functions identified as desirable. Penumbra combines tag-based policy specification and logic-based policy enforcement to support users' desired policies.

11.1 Summary of contributions

The first key contribution of this thesis are studies of users' needs and preferences for access control for personal data. The first study explores the topic broadly, providing high-level design guidelines across a variety of content types and possible sharing scenarios. The second and third studies examine specific potential access-control mechanisms in more detail, finding that both reactive policy creation and tag-based policy specification are promising approaches that fit with users' mental models and provide important benefits compared to more traditional policymaking approaches.

A second important contribution is the design, implementation, and evaluation of Penumbra, an access-control architecture designed to support users' needs. Penumbra uses tag-based policy specification to support users' fine-grained mental models for organizing and categorizing content, and uses logic-based policy enforcement to provide principled security while enabling diverse flexible, decentralized policies. In addition, Penumbra supports distributed file access, private tags, tag disagreement between users, and unforgeable audit records. Our implementation and evaluation demonstrate that realistic policies can be specified and enforced with limited overhead appropriate for interactive use cases.

A third contribution is the development of realistic case studies, drawn directly from our user studies, that capture real-world policies in detail. These case studies, which can be applied to other systems as well as Penumbra, are used to demonstrate that our policy language is sufficiently expressive to support realistic use cases. Based on these case studies, we develop a set of file-system traces that can be used to evaluate the performance of personal distributed file systems that support flexible access control.

Fourth, this thesis distills a set of requirements for usable access control for personal data. These requirements, which span the design space from security properties and policy mechanisms to user interfaces and policy inference, can be used to approach a broad variety of related problems, including but also beyond personal distributed file systems.

11.2 Further research directions

This section describes opportunities for future research building on my experience studying users' needs and preferences and developing Penumbra.

11.2.1 Improvements to Penumbra

There are several interesting research opportunities for directly extending Penumbra's functionality and improving its performance. First, for Penumbra to be practical, users must be able to acquire the credentials necessary to access content to which they are authorized. These may include delegation credentials (currently not protected in Penumbra) as well as tags that require proof of access. This raises the question of how we can decide which credentials (delegation or tags) should be proactively distributed across devices, to maximize the chances that proofs can be made immediately on demand without needing to first request credentials? At the same time, device-constrained resources and the principle of least privilege suggest that overprovisioning by prefetching as many credentials as possible is less than optimal. Progress in this area could build on work related to caching and prediction based on various kinds of data locality (e.g. [61, 104]). In addition, as discussed in Section 10.1.3, delegation credentials can themselves be highly sensitive, so algorithms for distributing credentials can build on the strong existing work in trust negotiation and zero-knowledge proofs (e.g., [96, 150]).

Second, Penumbra would be more usable if users who specify a too-broad query could receive partial answers. For example, a user who asks to list all photos but is only entitled to view public photos should receive the list of public photos, not no photos. To accomplish this would require developing a query restriction technique, in which the user's agent rewrites her metadata query, limiting it to that subset of the original request for which the agent can successfully prove authorized access. This can be analogized to the problem of

database query optimization (e.g. [57, 114]), but will require additional work to account for the lack of organized schema within the open user-defined tag environment Penumbra supports.

11.2.2 Access control beyond files

Our user studies, and consequently the design of Penumbra, support access-control primitives that we believe make sense for personal data in the context of files. Many current and evolving technologies, however, enable storing and sharing personal data that does not map well to a traditional file system, and managing access to these resources is increasingly important.

Important content can include sensor data from mobile devices and wearable computing technologies, household data generated as part of the so-called "internet of things," financial and medical records, and other digital content. Controlling access to this content poses several key challenges. First, the sheer amount of such data exacerbates the tension between users' complex policy goals and their limited time and ability to manage them.. In addition, while studies have shown that users have trouble accurately setting policy a priori (Chapters 3 and 4, [18]), at least users have a baseline ability to reason about sharing photos and documents. Managing policy for, to take one example, accelerometer data from a mobile phone is much more challenging, as users may not understand the associated risks or consequences. Handling these situations will require researchers to first understand the risks and consequences themselves, then to find ways to make these consequences clear to users so they can make informed decisions. improved tools for policy management will also be necessary.

From a system design perspective, mechanisms like tag-based policy and logic-based enforcement that can be sensible and reasonably efficient for managing traditional files may no longer make sense in the context of object and document stores or new database architectures. Designing for these systems will require working with users to understand the proper granularity at which access should be controlled, as well as which factors important to users' decision-making processes must be captured by policy primitives. From there, research will be needed to establish efficient mechanisms to implement these desired policy options.

11.2.3 Policy authoring and management

Penumbra provides an infrastructure layer that can support good abstractions and policy primitives for usable access control. A natural next step is to develop interfaces and tools to help users create and maintain policies, within Penumbra or other infrastructures.

This includes tools for visualizing and updating current policy, interfaces for enabling reactive and contextual policymaking, and mechanisms for supporting automated policy suggestion and inference. All of these areas are discussed in further detail in Section 10.3, as part of our recommendations for usable access control generally.

Bibliography

[1] FUSE: Filesystem in userspace. http://fuse.sourceforge.net. Cited on page 103.

[2] Average number of uploaded and linked photos of Facebook users as of January 2011, by gender. Statista, 2013. Cited on page 98.

[3] Martín Abadi, Michael Burrows, Butler Lampson, and Gordon Plotkin. A calculus for access control in distributed systems. *ACM TOPLAS*, 15(4):706–734, 1993. Cited on page 17.

[4] Mark S. Ackerman. The intellectual challenge of CSCW: The gap between social requirements and technical feasibility. *Human-Computer Interaction*, 15(2):179–203, 2000. Cited on page 9.

[5] Alessandro Acquisti and Ralph Gross. Imagined communities: Awareness, information sharing, and privacy on the Facebook. In *Proc. PETS*, 2006. Cited on page 7.

[6] Fabeah Adu-Oppong, Casey K. Gardiner, Apu Kapadia, and Patrick P. Tsang. Social circles: Tackling privacy in social networks. *Poster abstracts of SOUPS*, 2008. Cited on page 11.

[7] Yuvraj Agarwal and Malcolm Hall. ProtectMyPrivacy: Detecting and mitigating privacy leaks on iOS devices using crowdsourcing. In *Proc. ACM MobiSys*, 2013. Cited on page 12.

[8] Shane Ahern, Dean Eckles, Nathaniel S. Good, Simon King, Mor Naaman, and Rahul Nair. Over-exposed? Privacy patterns and considerations in online and mobile photo sharing. In *Proc. ACM CHI*, 2007. Cited on pages 7, 81, 98, and 114.

[9] Saleema Amershi, James Fogarty, and Daniel Weld. ReGroup: Interactive Machine Learning for On-Demand Group Creation in Social Networks. In *Proc. ACM CHI*, 2012. Cited on pages 11 and 118.

[10] Morgan Ames and Mor Naaman. Why we tag: motivations for annotation in mobile and online media. In *Proc. ACM CHI*, 2007. Cited on page 13.

[11] Mohd Anwar and Philip W. L. Fong. A visualization tool for evaluating access

control policies in Facebook-style social network systems. In *Proc. ACM SAC*, 2012. Cited on page 12.

[12] Andrew W. Appel and Edward W. Felten. Proof-carrying authentication. In *Proc. ACM CCS*, 1999. Cited on page 17.

[13] Apple. Apple iCloud. https://www.icloud.com, 2013. Cited on page 1.

[14] Ching-Man Au Yeung, Lalana Kagal, Nicholas Gibbins, and Nigel Shadbolt. Providing access control to online photo albums based on tags and linked data. In *Proc. AAAI-SSS:Social Semantic Web*, 2009. Cited on pages 14 and 61.

[15] Oshrat Ayalon and Eran . Retrospective privacy: managing longitudinal privacy in online social networks. In *Proc. SOUPS*, 2013. Cited on pages 7 and 81.

[16] Rebecca Balebako, Jaeyeon Jung, Wei Lu, Lorrie Faith Cranor, and Carolyn Nguyen. "little brothers watching you:" Raising awareness of data leaks on smartphones. In *Proc. SOUPS*, 2013. Cited on page 12.

[17] Dirk Balfanz, Drew Dean, and Mike Spreitzer. A security infrastructure for distributed Java applications. In *Proc. IEEE S & P*, 2000. Cited on page 18.

[18] Lujo Bauer, Lorrie Faith Cranor, Saranga Komanduri, Michelle L. Mazurek, Michael K. Reiter, Manya Sleeper, and Blase Ur. The post anachronism: The temporal dimension of Facebook privacy. In *Proc. ACM WPES*, 2013. Cited on pages 8 and 121.

[19] Lujo Bauer, Lorrie Faith Cranor, Robert W. Reeder, Michael K. Reiter, and Kami Vaniea. A user study of policy creation in a flexible access-control system. In *Proc. ACM CHI*, 2008. Cited on page 23.

[20] Lujo Bauer, Lorrie Faith Cranor, Michael K. Reiter, and Kami Vaniea. Lessons learned from the deployment of a smartphone-based access-control system. In *Proc. SOUPS*, 2007. Cited on page 6.

[21] Lujo Bauer, Scott Garriss, and Michael K. Reiter. Distributed proving in access-control systems. In *Proc. IEEE S&P*, 2005. Cited on pages 6, 17, 89, 105, and 117.

[22] Lujo Bauer, Scott Garriss, and Michael K. Reiter. Efficient proving for practical distributed access-control systems. In *Proc. ESORICS*, 2007. Cited on page 111.

[23] Lujo Bauer, Yuan Liang, Michael K. Reiter, and Chad Spensky. Discovering access-control misconfigurations: New approaches and evaluation methodologies. In *Proc. ACM CODASPY*, 2012. Cited on page 118.

[24] Lujo Bauer, Michael A. Schneider, and Edward W. Felten. A general and flexible access-control system for the Web. In *Proc. USENIX Security*, 2002. Cited on page

17.

[25] Moritz Y. Becker, Cédric Fournet, and Andrew D. Gordon. Design and semantics of a decentralized authorization language. In *Proc. IEEE CSF*, 2007. Cited on page 18.

[26] Moritz Y. Becker and Peter Sewell. Cassandra: Flexible trust management, applied to electronic health records. Cited on page 18.

[27] Matthias Beckerle and Leonardo A. Martucci. Formal definitions for usable access control rule sets from goals to metrics. In *Proc. SOUPS*, 2013. Cited on page 10.

[28] Alastair R. Beresford, Andrew Rice, Nicholas Skehin, and Ripduman Sohan. MockDroid: trading privacy for application functionality on smartphones. In *Proc. ACM HotMobile*, 2011. Cited on page 12.

[29] Andrew Besmer and Heather Richter Lipford. Moving beyond untagging: Photo privacy in a tagged world. In *Proc. ACM CHI*, 2010. Cited on pages 7, 98, and 114.

[30] Andrew Besmer, Jason Watson, and Heather Richter Lipford. The impact of social navigation on privacy policy configuration. In *Proc. SOUPS*, 2010. Cited on page 117.

[31] Konstantin Beznosov and Olga Beznosova. On the imbalance of the security problem space and its expected consequences. *Information Management & Computer Security*, 15(5):420–431, 2007. Cited on page 21.

[32] Matt Blaze, Joan Feigenbaum, and Martin Strauss. Compliance checking in the PolicyMaker trust management system. *Proc. Financial Cryptography*, 1998. Cited on page 18.

[33] Kevin D. Bowers, Lujo Bauer, Deepak Garg, Frank Pfenning, and Michael K. Reiter. Consumable credentials in logic-based access-control systems. In *Proc. NDSS*, 2007. Cited on page 116.

[34] A. J. Bernheim Brush and Kori Inkpen. Yours, mine and ours? Sharing and use of technology in domestic environments. In *Proc. ACM UbiComp*. 2007. Cited on pages 8 and 81.

[35] Kelly Caine and Rima Hanania. Patients want granular privacy control over health information in electronic medical records. *Journal of the American Medical Informatics Association*, 20(1):7–15, 2012. Cited on page 9.

[36] Xiang Cao and Lee Iverson. Intentional access management: Making access control usable for end-users. In *Proc. SOUPS*, 2006. Cited on page 10.

[37] Sunny Consolvo, Ian E. Smith, Tara Matthews, Anthony LaMarca, Jason Tabert, and Pauline Powledge. Location disclosure to social relations: why, when, & what people

want to share. In *Proc. ACM CHI*, 2005. Cited on pages 8 and 42.

[38] Facebook & your privacy: Who sees the data you share on the biggest social network? Consumer Reports Magazine, June 2012. Cited on page 1.

[39] David Coursey. Google apologizes for Buzz privacy issues. *PCWorld*, February 15, 2010. https://www.pcworld.com/article/189329/Google_Apologizes_for_Buzz_Privacy_Issues.html. Cited on page 1.

[40] Justin Cranshaw, Jonathan Mugan, and Norman Sadeh. User-controllable learning of location privacy policies with Gaussian mixture models. *Proc. AAAI*, 2011. Cited on pages 13 and 118.

[41] George Danezis. Inferring privacy policies for social networking services. In *Proc. ACM AISec*, 2009. Cited on page 11.

[42] Juri L. De Coi, Ekaterini Ioannou, Arne Koesling, Wolfgang Nejdl, and Daniel Olmedilla. Access control for sharing semantic data across desktops. In *Proc. ISWC*, 2007. Cited on page 14.

[43] Emiliano De Cristofaro, Claudio Soriente, Gene Tsudik, and Andrew Williams. Hummingbird: Privacy at the time of Twitter. In *Proc. IEEE S&P*, 2012. Cited on page 15.

[44] John DeTreville. Binder, a logic-based security language. In *Proc. IEEE S&P*, 2002. Cited on page 18.

[45] Paul Dourish, E. Grinter, Jessica Delgado de la Flor, and Melissa Joseph. Security in the wild: user strategies for managing security as an everyday, practical problem. *Personal Ubiquitous Comput.*, 8(6):391–401, 2004. Cited on page 1.

[46] W. Keith Edwards, Mark W. Newman, and Erika Shehan Poole. The infrastructure problem in HCI. In *Proc. ACM CHI*, 2010. Cited on page 2.

[47] Conal Elliott and Frank Pfenning. A semi-functional implementation of a higher-order logic programming language. In *Topics in Advanced Language Implementation*. MIT Press, 1991. Cited on page 105.

[48] Lujun Fang and Kristen LeFevre. Privacy wizards for social networking sites. In *Proc. WWW*, 2010. Cited on pages 11 and 118.

[49] Joan Feigenbaum and Angelos Keromytis. KeyNote: Trust management for public-key infrastructures. *Security Protocols*, 1999. Cited on page 18.

[50] Adrienne Porter Felt, Serge Egelman, Matthew Finifter, and Devdatta Akhawe. How to ask for permission. *Proc. USENIX HotSec*, 2012. Cited on page 12.

[51] Deepak Garg and Frank Pfenning. A proof-carrying file system. In *Proc. IEEE S&P*,

2010. Cited on pages 2, 18, and 116.

[52] Roxana Geambasu, Magdalena Balazinska, Steven D. Gribble, and Henry M. Levy. Homeviews: Peer-to-peer middleware for personal data sharing applications. In *Proc. ACM SIGMOD*, 2007. Cited on pages 17 and 28.

[53] David K. Gifford, Pierre Jouvelot, Mark A. Sheldon, and James W. O'Toole. Semantic file systems. In *Proc. ACM SOSP*, 1991. Cited on page 16.

[54] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proc. ACM CCS*, 2006. Cited on page 15.

[55] Manish Gupta, Rui Li, Zhijun Yin, and Jiawei Han. Survey on social tagging techniques. *SIGKDD Explorations Newsletter*, 2010. Cited on page 13.

[56] Yuri Gurevich and Itay Neeman. DKAL: Distributed-knowledge authorization language. In *Proc. IEEE CSF*, 2008. Cited on page 18.

[57] Alon Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, 2001. Cited on page 121.

[58] Michael Hart, Claude Castille, Rob Johnson, and Amanda Stent. Usable privacy controls for blogs. In *Proc. IEEE CSE*, 2009. Cited on pages 15 and 61.

[59] Don Hedeker. Mixed models for longitudinal ordinal and nominal outcomes, 2012. http://www.uic.edu/classes/bstt/bstt513/OrdNomLS.pdf. Cited on page 48.

[60] Alexander Heitzmann, Bernardo Palazzi, Charalampos Papamanthou, and Roberto Tamassia. Effective visualization of file system access-control. In *Proc. VizSec*, 2008. Cited on page 10.

[61] Brett D Higgins, Jason Flinn, T J Giuli, Brian Noble, Christopher Peplin, and David Watson. Informed mobile prefetching. In *Proc. ACM MobiSys*, 2012. Cited on page 120.

[62] Kashmir Hill. Teacher accidentally puts racy photo on students' iPad. School bizarrely suspends students. *Forbes*, October 18, 2012. http://www.forbes.com/sites/kashmirhill/2012/10/18/teacher-accidentally-puts-racy-photo-on-students-ipad-school-bizarrely-suspends-students/. Cited on page 1.

[63] Kashmir Hill. What college admission officers don't like seeing on Facebook: Vulgarity, drinking photos & illegal activities. *Forbes*, October 12, 2012. http://www.forbes.com/sites/kashmirhill/2012/10/12/what-college-admission-officers-dont-like-seeing-on-facebook-profiles-vulgarity-

drinking-photos-and-illegal-activities/. Cited on page 1.

[64] Jon Howell and Stuart Schechter. What you see is what they get. *Proc. IEEE W2SP*, 2010. Cited on page 12.

[65] Iulia Ion, Niharika Sachdeva, Ponnurangam Kumaraguru, and Srdjan Čapkun. Home is safer than the cloud! Privacy concerns for consumer cloud storage. In *Proc. SOUPS*, 2011. Cited on page 9.

[66] Maritza Johnson, Serge Egelman, and Steven M. Bellovin. Facebook and privacy: It's complicated. In *Proc. SOUPS*, 2012. Cited on pages 1, 7, and 98.

[67] Maritza Johnson, John Karat, Clare-Marie Karat, and Keith Grueneberg. Usable policy template authoring for iterative policy refinement. In *Proc. IEEE POLICY*, 2010. Cited on page 2.

[68] Maritza L. Johnson, Steven M. Bellovin, Robert W. Reeder, and Stuart E. Schechter. Laissez-faire file sharing: Access control designed for individuals at the endpoints. In *Proc. ACM NSPW*, 2009. Cited on page 10.

[69] Simon Jones and Eamonn O'Neill. Feasibility of structural network clustering for group-based privacy control in social networks. In *Proc. SOUPS*, 2010. Cited on page 11.

[70] Sanjay Kairam, Mike Brzozowski, David Huffaker, and Ed Chi. Talking in circles: Selective sharing in Google+. In *Proc. ACM CHI*, 2012. Cited on page 115.

[71] Apu Kapadia, Geetanjali Sampemane, and Roy H. Campbell. KNOW why your access was denied: Regulating feedback for usable security. In *Proc. ACM CCS*, 2004. Cited on page 115.

[72] Amy K. Karlson, A.J. Bernheim Brush, and Stuart Schechter. Can I borrow your phone? Understanding concerns when sharing mobile phones. In *Proc. ACM CHI*, 2009. Cited on pages 8, 38, and 81.

[73] Patrick Gage Kelley, Robin Brewer, Yael Mayer, Lorrie Faith Cranor, and Norman Sadeh. An investigation into Facebook friend grouping. In *Proc. IFIP NTERACT*, 2011. Cited on page 11.

[74] Patrick Gage Kelley, Lorrie Faith Cranor, and Norman Sadeh. Privacy as part of the app decision-making process. In *Proc. ACM CHI*, 2013. Cited on page 12.

[75] Patrick Gage Kelley, Paul Hankes Drielsma, Norman Sadeh, and Lorrie Faith Cranor. User-controllable learning of security and privacy policies. In *Proc. AISec*, 2008. Cited on page 118.

[76] Angelos D. Keromytis, Sotiris Ioannidis, Michael B. Greenwald, and Jonathan M.

Smith. The STRONGMAN architecture. In *Proc. DARPA Information Survivability Conference and Exposition*, 2003. Cited on page 18.

[77] Tiffany Hyun-Jin Kim, Lujo Bauer, James Newsome, Adrian Perrig, and Jesse Walker. Challenges in access right assignment for secure home networks. In *Proc. USENIX HotSec*, 2010. Cited on pages 115, 116, and 117.

[78] Peter Klemperer, Yuan Liang, Michelle L. Mazurek, Manya Sleeper, Blase Ur, Lujo Bauer, Lorrie Faith Cranor, Nitin Gupta, and Michael K. Reiter. Tag, you can see it! Using tags for access control in photo sharing. In *Proc. ACM CHI*, 2012. Cited on pages 4 and 110.

[79] Vivek Krishnan, Mahesh V Tripunitara, Kinson Chik, and Tony Bergstrom. Relating declarative semantics and usability in access control. In *Proc. SOUPS*, 2012. Cited on page 11.

[80] Butler Lampson, Martín Abadi, Michael Burrows, and Edward Wobber. Authentication in distributed systems: Theory and practice. *ACM Trans. Comput. Syst.*, 10(4):265–310, 1992. Cited on page 86.

[81] Vassilios Lekakis, Yunus Basagalar, and Pete Keleher. Don't trust your roommate, or access control and replication protocols in "home" environments. In *Proc. USENIX HotOS*, 2012. Cited on page 17.

[82] Chris Lesniewski-Laas, Bryan Ford, Jacob Strauss, Robert Morris, and M. Frans Kaashoek. Alpaca: Extensible authorization for distributed services. In *Proc. ACM CCS*, 2007. Cited on page 17.

[83] Ninghui Li, John C Mitchell, and William H. Winsborough. Design of a role-based trust-management framework. In *Proc. IEEE S&P*, 2002. Cited on pages 2 and 18.

[84] Linda Little, Elizabeth Sillence, and Pam Briggs. Ubiquitous systems and the family: Thoughts about the networked home. In *Proc. SOUPS*, 2009. Cited on page 8.

[85] Bin Liu, Jialiu Lin, and Norman Sadeh. Reconciling mobile app privacy and usability on smartphones: Could user privacy profiles help? In *Proc. WWW*, 2014. Cited on page 12.

[86] Catherine C. Marshall. Toward an ecology of hypertext annotation. In *Proc. HYPERTEXT*, 1998. Cited on page 13.

[87] Amirreza Masoumzadeh and James Joshi. Privacy settings in social networking systems: What you cannot control. In *Proc. ACM ASIACCS*, 2013. Cited on page 1.

[88] Michelle L. Mazurek, J. P. Arsenault, Joanna Bresee, Nitin Gupta, Iulia Ion, Christina Johns, Daniel Lee, Yuan Liang, Jenny Olsen, Brandon Salmon, Richard Shay, Kami

Vaniea, Lujo Bauer, Lorrie Faith Cranor, Gregory R. Ganger, and Michael K. Reiter. Access control for home data sharing: Attitudes, needs and practices. In *Proc. ACM CHI*, 2010. Cited on page 4.

[89] Michelle L. Mazurek, Peter F. Klemperer, Richard Shay, Hassan Takabi, Lujo Bauer, and Lorrie Faith Cranor. Exploring reactive access control. In *Proc. ACM CHI*, 2011. Cited on page 4.

[90] Michelle L. Mazurek, Yuan Liang, William Melicher, Manya Sleeper, Lujo Bauer, Gregory R Ganger, Nitin Gupta, and Michael K. Reiter. Toward strong, usable access control for shared distributed data. In *FAST'14: Proceedings of the 12th USENIX conference on File and Storage Technologies*, 2014. Cited on page 4.

[91] Alessandra Mazzia, Kristen LeFevre, and Eytan Adar. The PViz comprehension tool for social network privacy settings. In *Proc. SOUPS*, 2012. Cited on page 11.

[92] Daniel D. McCracken and Rosalee Jean Wolfe. *User-centered website development: A human-computer interaction approach*. Prentice Hall Englewood Cliffs, 2004. Cited on page 97.

[93] Microsoft. Windows SkyDrive. http://windows.microsoft.com/en-us/skydrive, 2013. Cited on page 1.

[94] Andrew D. Miller and W. Keith Edwards. Give and take: a study of consumer photo-sharing culture and practice. In *Proc. ACM CHI*, 2007. Cited on page 62.

[95] Jun-Ki Min, Jason Wiese, Jason I. Hong, and John Zimmerman. Mining smartphone data to classify life-facets of social relationships. In *Proc. ACM CSCW*, 2013. Cited on page 118.

[96] Kazuhiro Minami, Nikita Borisov, Marianne Winslett, and Adam J. Lee. Confidentiality-preserving proof theories for distributed proof systems. In *Proc. ACM ASIACCS*, 2011. Cited on pages 115 and 120.

[97] Tom Mitchell. *Machine Learning*. McGraw-Hill, 1997. Cited on page 67.

[98] Mohammad Nauman, Sohail Khan, and Xinwen Zhang. Apex: Extending Android permission model and enforcement with user-defined runtime constraints. In *Proc. ACM ASIACCS*, 2010. Cited on page 12.

[99] Rafe Needleman. How to fix Facebook's new privacy settings. *cnet*, December 10, 2009. http://news.cnet.com/8301-19882_3-10413317-250.html. Cited on page 1.

[100] Jakob Nielsen. *Usability engineering*. Morgan Kaufmann, 1993. Cited on pages 3 and 106.

[101] Oded Nov, Mor Naaman, and Chen Ye. What drives content tagging: the case of photos on Flickr. In *Proc. ACM CHI*, 2008. Cited on page 14.

[102] Judith S. Olson, Jonathan Grudin, and Eric Horvitz. A study of preferences for sharing and privacy. In *Proc. ACM CHI EA*, 2005. Cited on pages 5, 98, and 116.

[103] Sameer Patil, Greg Norcie, Apu Kapadia, and Adam J Lee. Reasons, rewards, regrets: Privacy considerations in location sharing as an interactive practice. In *Proc. SOUPS*, 2012. Cited on page 9.

[104] R. Hugo Patterson, Garth A. Gibson, Eka Ginting, Daniel Stodolsky, and Jim Zelenka. Informed prefetching and caching. In *Proc. ACM SOSP*, 1995. Cited on page 120.

[105] Daniel Peek and Jason Flinn. EnsemBlue: Integrating distributed storage and consumer electronics. In *Proc. USENIX OSDI*, 2006. Cited on pages 16 and 97.

[106] Ansley Post, Petr Kuznetsov, and Peter Druschel. PodBase: Transparent storage management for personal devices. In *Proc. IPTPS*, 2008. Cited on page 1.

[107] Dean Povey. Optimistic security: A new access control paradigm. In *Proc. ACM NSPW*, 1999. Cited on page 12.

[108] Venugopalan Ramasubramanian, Thomas L. Rodeheffer, Douglas B. Terry, Meg Walraed-Sullivan, Ted Wobber, Catherine C. Marshall, and Amin Vahdat. Cimbiosys: A platform for content-based partial replication. In *Proc. NSDI*, 2009. Cited on pages 1 and 105.

[109] Maryam N. Razavi and Lee Iverson. A grounded theory of information sharing behavior in a personal learning space. In *Proc. ACM CSCW*, 2006. Cited on pages 6, 51, 81, 98, and 117.

[110] Robert W. Reeder, Lujo Bauer, Lorrie Faith Cranor, Michael K. Reiter, Kelli Bacon, Keisha How, and Heather Strong. Expandable grids for visualizing and authoring computer security policies. In *Proc. ACM CHI*, 2008. Cited on pages 2, 10, 78, and 117.

[111] Lyn Richards. *Handling Qualitative Data: A Practical Guide*. Sage Publications, 2007. Cited on pages 23 and 46.

[112] Lyn Richards and Janice M. Morse. *Readme First for a User's Guide to Qualitative Methods*. Sage Publications, 2007. Cited on pages 23 and 46.

[113] Oriana Riva, Qin Yin, Dejan Juric, Ercan Ucan, and Timothy Roscoe. Policy expressivity in the Anzere personal cloud. In *Proc. ACM SOCC*, 2011. Cited on pages 1, 16, and 97.

[114] Shariq Rizvi, Alberto Mendelzon, S. Sudarshan, and Prasan Roy. Extending query

rewriting techniques for fine-grained access control. In *Proc. ACM SIGMOD*, 2004. Cited on page 121.

[115] Franziska Roesner, Tadayoshi Kohno, Alexander Moshchuk, Bryan Parno, Helen J. Wang, and Crispin Cowan. User-driven access control: Rethinking permission granting in modern operating systems. In *Proc. IEEE S&P*, 2012. Cited on page 12.

[116] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *Proc. EUROCRYPT*. 2005. Cited on page 15.

[117] Brandon Salmon, Steven W. Schlosser, Lorrie Faith Cranor, and Gregory R. Ganger. Perspective: Semantic data management for the home. In *Proc. USENIX FAST*, 2009. Cited on pages 16, 61, 81, 105, and 116.

[118] Fred B. Schneider, Kevin Walsh, and Emin Gün Sirer. Nexus authorization logic (NAL):Design rationale and applications. *ACM Trans. Inf. Syst. Secur.*, 14(1):1–28, 2011. Cited on page 18.

[119] Stan Schroeder. Facebook privacy: 10 settings every user needs to know. *Mashable*, February 7, 2011. http://mashable.com/2011/02/07/facebook-privacy-guide/. Cited on page 1.

[120] Margo Seltzer and Nicholas Murphy. Hierarchical file systems are dead. In *Proc. USENIX HotOS*, 2009. Cited on pages 16, 81, and 116.

[121] Shilad Sen, Shyong K. Lam, Al Mamunur Rashid, Dan Cosley, Dan Frankowski, Jeremy Osterhouse, F. Maxwell Harper, and John Riedl. tagging, communities, vocabulary, evolution. In *Proc. ACM CSCW*, 2006. Cited on page 14.

[122] Börkur Sigurbjörnsson and Roelof van Zwol. Flickr tag recommendation based on collective knowledge. In *Proc. WWW*, 2008. Cited on page 78.

[123] Manya Sleeper, Justin Cranshaw, Patrick Gage Kelley, Blase Ur, Alessandro Acquisti, Lorrie Faith Cranor, and Norman Sadeh. "I read my Twitter the next morning and was astonished": A conversational perspective on Twitter regrets. In *Proc. ACM CHI*, 2013. Cited on page 117.

[124] Diana K. Smetters and Nathan Good. How users use access control. In *Proc. SOUPS*, 2009. Cited on pages 6, 98, and 116.

[125] Anna C. Squicciarini, Smitha Sundareswaran, Dan Lin, and Josh Wede. A3P: adaptive policy prediction for shared images over popular content sharing sites. In *Proc. Hypertext and Hypermedia*, 2011. Cited on pages 16, 67, and 77.

[126] Jessica Staddon, Philippe Golle, Martin Gagné, and Paul Rasmussen. A content-driven access control system. In *Proc. IDTrust*, 2008. Cited on pages 15 and 114.

[127] Jessica Staddon, David Huffaker, Larkin Brown, and Aaron Sedley. Are privacy concerns a turn-off? Engagement and privacy in social networks. In *Proc. SOUPS*, 2012. Cited on page 7.

[128] Jacob Strauss, Justin Mazzola Paluska, Chris Lesniewski-Laas, Bryan Ford, Robert Morris, and Frans Kaashoek. Eyo: device-transparent personal storage. In *Proc. USENIX ATC*, 2011. Cited on pages 1, 16, and 97.

[129] Markus Strohmaier, Christian Körner, and Roman Kern. Why do users tag? Detecting users' motivation for tagging in social tagging systems. *Proc. ICWSM*, 2010. Cited on page 14.

[130] Fred Stutzman, Ralph Gross, and Alessandro Acquisti. Silent listeners: The evolution of privacy and disclosure on Facebook. *Journal of Privacy and Confidentiality*, 4(2):2, 2013. Cited on page 81.

[131] Christopher Thompson, Maritza Johnson, Serge Egelman, David Wagner, and Jennifer King. When it's better to ask forgiveness than get permission: attribution mechanisms for smartphone resources. In *Proc. SOUPS*, 2013. Cited on page 12.

[132] Eran Toch, Justin Cranshaw, Paul Hankes Drielsma, Janice Y. Tsai, Patrick Gage Kelley, James Springfield, Lorrie Faith Cranor, Jason Hong, and Norman Sadeh. Empirical models of privacy in location sharing. In *Proc. ACM Ubicomp*, 2010. Cited on page 13.

[133] Janice Y. Tsai, Patrick Kelley, Paul Hankes Drielsma, Lorrie Faith Cranor, Jason Hong, and Norman Sadeh. Who's viewed you? The impact of feedback in a mobile location-sharing application. In *Proc. ACM CHI*, 2009. Cited on page 12.

[134] Kami Vaniea, Lujo Bauer, Lorrie Faith Cranor, and Michael K. Reiter. Out of sight, out of mind: Effects of displaying access-control information near the item it controls. In *Proc. IEEE PST*, 2012. Cited on pages 2 and 10.

[135] Jeffrey A. Vaughan, Limin Jia, Karl Mazurak, and Steve Zdancewic. Evidence-based audit. *Proc. IEEE CSF*, 2008. Cited on pages 19 and 117.

[136] Stephen Voida, W. Keith Edwards, Mark W. Newman, Rebecca E. Grinter, and Nicolas Ducheneaut. Share and share alike: exploring the user interface affordances of file sharing. In *Proc. ACM CHI*, 2006. Cited on page 6.

[137] Nitya Vyas, Anna C. Squicciarini, Chih-Cheng Chang, and Danfeng Yao. Towards automatic privacy management in Web 2.0 with semantic analysis on annotations. In *Proc. IEEE ColloborateCom*, 2009. Cited on pages 15, 67, and 77.

[138] Yang Wang, Armen Aghasaryan, Arvind Shrihari, David Pergament, Guy-Bertrand

Kamga, and Stéphane Betgé-Brezetz. Intelligent reactive access control for moving user data. In *Proc. PASSAT*, 2011. Cited on page 117.

[139] Yang Wang, Gregory Norcie, Saranga Komanduri, Alessandro Acquisti, Pedro Giovanni Leon, and Lorrie Faith Cranor. "I regretted the minute I pressed share": a qualitative study of regrets on Facebook. In *Proc. SOUPS*, 2011. Cited on page 117.

[140] Jason Watson, Michael Whitney, and Heather Richter Lipford. Configuring audience-oriented privacy policies. In *Proc. SafeConfig*, 2009. Cited on page 117.

[141] Ben Weitzenkorn. McAfee's rookie mistake gives away his location. *Scientific American*, December 4, 2012. https://www.scientificamerican.com/article.cfm?id=mcafees-rookie-mistake. Cited on pages 1 and 114.

[142] D. Wetherall, D. Choffnes, B. Greenstein, S. Han, P. Hornyack, J. Jung, S. Schechter, and X. Wang. Privacy revelations for web and mobile apps. In *Proc. USENIX HotOS*, 2011. Cited on page 12.

[143] Steve Whittaker, Ofer Bergman, and Paul Clough. Easy on that trigger dad: a study of long term family photo retrieval. *Personal and Ubiquitous Computing*, 14(1):31–43, 2010. Cited on pages 98 and 106.

[144] Jeannette M. Wing. A symbiotic relationship between formal methods and security. In *Proc. Workshops on Computer Security, Dependability and Assurance: From Needs to Solutions*, 1998. Cited on page 114.

[145] Pamela J. Wisniewski, Heather Richter Lipford, and David C. Wilson. Fighting for my space: Coping mechanisms for SNS boundary regulation. In *Proc. ACM CHI*, 2012. Cited on pages 7 and 115.

[146] Edward Wobber, Martín Abadi, Michael Burrows, and Butler Lampson. Authentication in the Taos operating system. In *Proc. ACM SOSP*, 1993. Cited on page 17.

[147] Ted Wobber, Thomas L. Rodeheffer, and Douglas B. Terry. Policy-based access control for weakly consistent replication. In *Proc. EuroSys*, 2010. Cited on pages 17 and 97.

[148] Zhongyuan Xu and Scott D. Stoller. Mining attribute-based access control policies from RBAC policies. In *Proc. CEWIT*, 2013. Cited on page 118.

[149] Sarita Yardi and Amy Bruckman. Income, race, and class: Exploring socioeconomic differences in family technology use. In *Proc. ACM CHI*, 2012. Cited on page 81.

[150] Ting Yu, Marianne Winslett, and Kent E. Seamons. Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. *Transactions on Information and System Security (TISSEC)*, 6(1):1–42, 2003. Cited

on pages 115 and 120.

[151] Xuan Zhao, Niloufar Salehi, Sasha Naranjit, Sara Alwaalan, Voida Stephen, and Dan Cosley. The many faces of Facebook: Experiencing social media as performance, exhibition, and personal archive. In *Proc. ACM CHI*, 2013. Cited on page 117.

[152] Gottfried Zimmermann and Gregg Vanderheiden. Accessible design and testing in the application development process: Considerations for an integrated approach. *Universal Access in the Information Society*, 7(1-2):117–128, 2008. Cited on pages 97 and 98.

[153] Arkaitz Zubiaga, Christian Körner, and Markus Strohmaier. Tags vs shelves: from social tagging to social classification. In *Proc. Hypertext and Hypermedia*, 2011. Cited on page 14.

A | Policy credentials for the case studies

This appendix details the policy credentials created for two of our five case studies, expressed in the Penumbra logic.

A.1 Susie

$K_{extdrive}$ signed $\forall\ k\ a\ v\ f$ (deleg $K_{extdrive}$ (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{extdrive}$ (key k) (actionRMD (attr $K_{extdrive}\ a\ v$) :: nil (file f) rmd))

K_{susie} signed $\forall\ k\ a\ v\ f$ (deleg K_{susie} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{susie} (key k) (actionRMD (attr $K_{extdrive}\ a\ v$) :: nil (file f) rmd))

K_{susie} signed $\forall\ k\ a\ v\ f\ o$ (deleg $K_{susie}\ K_{o.k}$ (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{susie}\ K_{o.k}$ (actionRMD (attr $K_{extdrive}\ a\ v$) :: nil (file f) rmd))

$K_{extdrive}$ signed deleg $K_{extdrive}\ K_{susie}$ (actionD $K_{extdrive}$ cf)

$K_{extdrive}$ signed deleg $K_{extdrive}\ K_{susie}$ (actionD $K_{extdrive}$ cmd)

$K_{extdrive}$ signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg $K_{extdrive}$ (key k) (actionRMD q1 (file f) rmd))

$K_{extdrive}$ signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg $K_{extdrive}$ (key k) (actionRMD q1 (file f) dmd))

$K_{extdrive}$ signed $\forall\ f\ k\ a\ v$ (attrForm $K_{extdrive}$ "creator" k (file f))
$\longrightarrow$ (deleg $K_{extdrive}$ (key k) (actionRMD (attr $K_{extdrive}\ a\ v$) :: nil (file f) rmd))

$K_{extdrive}$ signed $\forall\ f\ k$ (attrForm $K_{extdrive}$ "creator" k (file f))
$\longrightarrow$ (deleg $K_{extdrive}$ (key k) (actionF (file f) rf))

$K_{extdrive}$ signed $\forall\ f\ k$ (attrForm $K_{extdrive}$ "creator" k (file f))
$\longrightarrow$ (deleg $K_{extdrive}$ (key k) (actionF (file f) wf))

K_{phone} signed $\forall\ k\ a\ v\ f$ (deleg K_{phone} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{phone} (key k) (actionRMD (attr $K_{phone}\ a\ v$) :: nil (file f) rmd))

K_{susie} signed $\forall\ k\ a\ v\ f$ (deleg K_{susie} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{susie} (key k) (actionRMD (attr $K_{phone}\ a\ v$) :: nil (file f) rmd))

K_{susie} signed $\forall\ k\ a\ v\ f\ o$ (deleg $K_{susie}\ K_{o.k}$ (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{susie}\ K_{o.k}$ (actionRMD (attr $K_{phone}\ a\ v$) :: nil (file f) rmd))

K_{phone} signed deleg $K_{phone}\ K_{susie}$ (actionD K_{phone} cf)

K_{phone} signed deleg $K_{phone}\ K_{susie}$ (actionD K_{phone} cmd)

K_{phone} signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{phone} (key k) (actionRMD q1 (file f) rmd))

K_{phone} signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{phone} (key k) (actionRMD q1 (file f) dmd))

K_{phone} signed $\forall\ f\ k\ a\ v$ (attrForm K_{phone} "creator" k (file f))
$\longrightarrow$ (deleg K_{phone} (key k) (actionRMD (attr $K_{phone}\ a\ v$) :: nil (file f) rmd))

K_{phone} signed $\forall\ f\ k$ (attrForm K_{phone} "creator" k (file f))
$\longrightarrow$ (deleg K_{phone} (key k) (actionF (file f) rf))

K_{phone} signed $\forall\ f\ k$ (attrForm K_{phone} "creator" k (file f))
$\longrightarrow$ (deleg K_{phone} (key k) (actionF (file f) wf))

K_{laptop} signed $\forall\ k\ a\ v\ f$ (deleg K_{laptop} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{laptop} (key k) (actionRMD (attr $K_{laptop}\ a\ v$) :: nil (file f) rmd))

K_{susie} signed $\forall\ k\ a\ v\ f$ (deleg K_{susie} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{susie} (key k) (actionRMD (attr $K_{laptop}\ a\ v$) :: nil (file f) rmd))

K_{susie} signed $\forall\ k\ a\ v\ f\ o$ (deleg $K_{susie}\ K_{o.k}$ (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{susie}\ K_{o.k}$ (actionRMD (attr $K_{laptop}\ a\ v$) :: nil (file f) rmd))

K_{laptop} signed deleg $K_{laptop}\ K_{susie}$ (actionD K_{laptop} cf)

K_{laptop} signed deleg $K_{laptop}\ K_{susie}$ (actionD K_{laptop} cmd)

K_{laptop} signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{laptop} (key k) (actionRMD q1 (file f) rmd))

K_{laptop} signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{laptop} (key k) (actionRMD q1 (file f) dmd))

K_{laptop} signed $\forall\ f\ k\ a\ v$ (attrForm K_{laptop} "creator" k (file f))

$\longrightarrow$ (deleg K_{laptop} (key k) (actionRMD (attr K_{laptop} a v) :: nil (file f) rmd))

K_{laptop} signed $\forall$ f k (attrForm K_{laptop} "creator" k (file f))
$\longrightarrow$ (deleg K_{laptop} (key k) (actionF (file f) rf))

K_{laptop} signed $\forall$ f k (attrForm K_{laptop} "creator" k (file f))
$\longrightarrow$ (deleg K_{laptop} (key k) (actionF (file f) wf))

K_{cloud} signed $\forall$ k a v f (deleg K_{cloud} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{cloud} (key k) (actionRMD (attr K_{cloud} a v) :: nil (file f) rmd))

K_{susie} signed $\forall$ k a v f (deleg K_{susie} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{susie} (key k) (actionRMD (attr K_{cloud} a v) :: nil (file f) rmd))

K_{susie} signed $\forall$ k a v f o (deleg K_{susie} $K_{o.k}$ (actionF (file f) sf))
$\longrightarrow$ (deleg K_{susie} $K_{o.k}$ (actionRMD (attr K_{cloud} a v) :: nil (file f) rmd))

K_{cloud} signed deleg K_{cloud} K_{susie} (actionD K_{cloud} cf)

K_{cloud} signed deleg K_{cloud} K_{susie} (actionD K_{cloud} cmd)

K_{cloud} signed $\forall$ $q1$ k f (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{cloud} (key k) (actionRMD q1 (file f) rmd))

K_{cloud} signed $\forall$ $q1$ k f (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{cloud} (key k) (actionRMD q1 (file f) dmd))

K_{cloud} signed $\forall$ f k a v (attrForm K_{cloud} "creator" k (file f))
$\longrightarrow$ (deleg K_{cloud} (key k) (actionRMD (attr K_{cloud} a v) :: nil (file f) rmd))

K_{cloud} signed $\forall$ f k (attrForm K_{cloud} "creator" k (file f))
$\longrightarrow$ (deleg K_{cloud} (key k) (actionF (file f) rf))

K_{cloud} signed $\forall$ f k (attrForm K_{cloud} "creator" k (file f))
$\longrightarrow$ (deleg K_{cloud} (key k) (actionF (file f) wf))

$K_{extdrive}$ signed $\forall$ f (deleg $K_{extdrive}$ K_{susie} (actionF (file f) sf))

$K_{extdrive}$ signed $\forall$ f (deleg $K_{extdrive}$ K_{susie} (actionF (file f) rf))

$K_{extdrive}$ signed $\forall$ f (deleg $K_{extdrive}$ K_{susie} (actionF (file f) wf))

$K_{extdrive}$ signed $\forall$ q f (deleg $K_{extdrive}$ K_{susie} (actionRMD q (file f) rmd))

$K_{extdrive}$ signed $\forall$ q f (deleg $K_{extdrive}$ K_{susie} (actionRMD q (file f) dmd))

K_{phone} signed $\forall\ f$ (deleg K_{phone} K_{susie} (actionF (file f) sf))

K_{phone} signed $\forall\ f$ (deleg K_{phone} K_{susie} (actionF (file f) rf))

K_{phone} signed $\forall\ f$ (deleg K_{phone} K_{susie} (actionF (file f) wf))

K_{phone} signed $\forall\ q\ f$ (deleg K_{phone} K_{susie} (actionRMD q (file f) rmd))

K_{phone} signed $\forall\ q\ f$ (deleg K_{phone} K_{susie} (actionRMD q (file f) dmd))

K_{laptop} signed $\forall\ f$ (deleg K_{laptop} K_{susie} (actionF (file f) sf))

K_{laptop} signed $\forall\ f$ (deleg K_{laptop} K_{susie} (actionF (file f) rf))

K_{laptop} signed $\forall\ f$ (deleg K_{laptop} K_{susie} (actionF (file f) wf))

K_{laptop} signed $\forall\ q\ f$ (deleg K_{laptop} K_{susie} (actionRMD q (file f) rmd))

K_{laptop} signed $\forall\ q\ f$ (deleg K_{laptop} K_{susie} (actionRMD q (file f) dmd))

K_{cloud} signed $\forall\ f$ (deleg K_{cloud} K_{susie} (actionF (file f) sf))

K_{cloud} signed $\forall\ f$ (deleg K_{cloud} K_{susie} (actionF (file f) rf))

K_{cloud} signed $\forall\ f$ (deleg K_{cloud} K_{susie} (actionF (file f) wf))

K_{cloud} signed $\forall\ q\ f$ (deleg K_{cloud} K_{susie} (actionRMD q (file f) rmd))

K_{cloud} signed $\forall\ q\ f$ (deleg K_{cloud} K_{susie} (actionRMD q (file f) dmd))

K_{susie} signed K_{laptop} speaksfor $K_{susie.trustedDevices}$

K_{susie} signed K_{phone} speaksfor $K_{susie.trustedDevices}$

K_{susie} signed K_{cloud} speaksfor $K_{susie.trustedDevices}$

K_{susie} signed $K_{extdrive}$ speaksfor $K_{susie.trustedDevices}$

K_{susie} signed $K_{susie.trustedDevices}$ speaksfor K_{susie}

K_{susie} signed $K_{exteacher1}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher10}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher11}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher12}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher13}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher14}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher2}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher3}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher4}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher5}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher6}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher7}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher8}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{exteacher9}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed $K_{friendboyfriend1}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend10}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend11}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend12}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend13}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend2}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend3}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend4}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend5}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend6}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend7}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend8}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{friendboyfriend9}$ speaksfor $K_{susie.public}$

K_{susie} signed $K_{neighbor1}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor10}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor11}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor12}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor13}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor2}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor3}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor4}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor5}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor6}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor7}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor8}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{neighbor9}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{roommate1}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate10}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate11}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate12}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate13}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate2}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate3}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate4}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate5}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate6}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate7}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate8}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{roommate9}$ speaksfor $K_{susie.roommates}$

K_{susie} signed $K_{parentfriend}$ speaksfor $K_{susie.oldpeople}$

K_{susie} signed K_{dad} speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{acquaintance}$ speaksfor $K_{susie.notclose}$

K_{susie} signed $K_{friendfriend}$ speaksfor $K_{susie.notclose}$

K_{susie} signed K_{boss} speaksfor $K_{susie.public}$

K_{susie} signed $\forall f$ (attrForm K_{susie} "type" "photo" (file f))
$\longrightarrow$ (deleg K_{susie} $K_{susie.roommates}$ (actionF (file f) rf))

K_{susie} signed $\forall f$ (deleg K_{susie} $K_{susie.roommates}$ (actionRMD (attr K_{susie} "type" "photo") :: nil (file f) rmd))

K_{susie} signed $\forall f$ (attrForm K_{susie} "type" "photo" (file f))
$\longrightarrow$ (deleg K_{susie} $K_{susie.roommates}$ (actionF (file f) sf))

K_{susie} signed $\forall f$ (attrForm K_{susie} "type" "photo" (file f)) $\wedge$ (attrForm K_{susie} "mom_sensitive" "No" (file f))
$\longrightarrow$ (deleg K_{susie} K_{mom} (actionF (file f) rf))

K_{susie} signed $\forall f$ (deleg K_{susie} K_{mom} (actionRMD (attr K_{susie} "mom_sensitive" "No") :: (attr K_{susie} "type" "photo") :: nil (file f) rmd))

K_{susie} signed $\forall f$ (attrForm K_{susie} "type" "photo" (file f)) $\wedge$ (attrForm K_{susie} "mom_sensitive" "No" (file f))
$\longrightarrow$ (deleg K_{susie} K_{mom} (actionF (file f) sf))

K_{susie} signed $\forall f$ (attrForm K_{susie} "personal" "No" (file f)) $\wedge$ (attrForm K_{susie} "type" "photo" (file f)) $\wedge$ (attrForm K_{susie} "red_flag" "No" (file f)) $\wedge$ (attrForm K_{susie} "very_personal" "No" (file f))
$\longrightarrow$ (deleg K_{susie} $K_{susie.notclose}$ (actionF (file f) rf))

K_{susie} signed $\forall f$ (deleg K_{susie} $K_{susie.notclose}$ (actionRMD (attr K_{susie} "very_personal" "No") :: (attr K_{susie} "red_flag" "No") :: (attr K_{susie} "type" "photo") :: (attr K_{susie} "personal" "No") :: nil (file f) rmd))

K_{susie} signed $\forall f$ (attrForm K_{susie} "personal" "No" (file f)) $\wedge$ (attrForm K_{susie} "type" "photo" (file f)) $\wedge$ (attrForm K_{susie} "red_flag" "No" (file f)) $\wedge$ (attrForm K_{susie} "very_personal" "No" (file f))

$\longrightarrow$ (deleg K_{susie} $K_{susie.notclose}$ (actionF (file f) sf))

K_{susie} signed $\forall$ f (attrForm K_{susie} "type" "photo" (file f)) $\wedge$ (attrForm K_{susie} "red_flag" "No" (file f))
$\longrightarrow$ (deleg K_{susie} $K_{susie.oldpeople}$ (actionF (file f) rf))

K_{susie} signed $\forall$ f (deleg K_{susie} $K_{susie.oldpeople}$ (actionRMD (attr K_{susie} "red_flag" "No") :: (attr K_{susie} "type" "photo") :: nil (file f) rmd))

K_{susie} signed $\forall$ f (attrForm K_{susie} "type" "photo" (file f)) $\wedge$ (attrForm K_{susie} "red_flag" "No" (file f))
$\longrightarrow$ (deleg K_{susie} $K_{susie.oldpeople}$ (actionF (file f) sf))

K_{susie} signed $\forall$ f (attrForm K_{susie} "personal" "No" (file f)) $\wedge$ (attrForm K_{susie} "kids" "No" (file f)) $\wedge$ (attrForm K_{susie} "type" "photo" (file f)) $\wedge$ (attrForm K_{susie} "red_flag" "No" (file f)) $\wedge$ (attrForm K_{susie} "very_personal" "No" (file f))
$\longrightarrow$ (deleg K_{susie} $K_{susie.public}$ (actionF (file f) rf))

K_{susie} signed $\forall$ f (deleg K_{susie} $K_{susie.public}$ (actionRMD (attr K_{susie} "very_personal" "No") :: (attr K_{susie} "red_flag" "No") :: (attr K_{susie} "type" "photo") :: (attr K_{susie} "kids" "No") :: (attr K_{susie} "personal" "No") :: nil (file f) rmd))

K_{susie} signed $\forall$ f (attrForm K_{susie} "personal" "No" (file f)) $\wedge$ (attrForm K_{susie} "kids" "No" (file f)) $\wedge$ (attrForm K_{susie} "type" "photo" (file f)) $\wedge$ (attrForm K_{susie} "red_flag" "No" (file f)) $\wedge$ (attrForm K_{susie} "very_personal" "No" (file f))
$\longrightarrow$ (deleg K_{susie} $K_{susie.public}$ (actionF (file f) sf))

A.2 Heather and Matt

K_{phone} signed $\forall$ k a v f (deleg K_{phone} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{phone} (key k) (actionRMD (attr K_{phone} a v) :: nil (file f) rmd))

$K_{heather}$ signed $\forall$ k a v f (deleg $K_{heather}$ (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{heather}$ (key k) (actionRMD (attr K_{phone} a v) :: nil (file f) rmd))

$K_{heather}$ signed $\forall$ k a v f o (deleg $K_{heather}$ $K_{o.k}$ (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{heather}$ $K_{o.k}$ (actionRMD (attr K_{phone} a v) :: nil (file f) rmd))

K_{phone} signed deleg K_{phone} $K_{heather}$ (actionD K_{phone} cf)

K_{phone} signed deleg K_{phone} $K_{heather}$ (actionD K_{phone} cmd)

K_{phone} signed $\forall$ $q1$ k f (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{phone} (key k) (actionRMD q1 (file f) rmd))

K_{phone} signed $\forall$ *q1* k f (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{phone} (key k) (actionRMD q1 (file f) dmd))

K_{phone} signed $\forall$ f k a v (attrForm K_{phone} "creator" k (file f))
$\longrightarrow$ (deleg K_{phone} (key k) (actionRMD (attr K_{phone} a v) :: nil (file f) rmd))

K_{phone} signed $\forall$ f k (attrForm K_{phone} "creator" k (file f))
$\longrightarrow$ (deleg K_{phone} (key k) (actionF (file f) rf))

K_{phone} signed $\forall$ f k (attrForm K_{phone} "creator" k (file f))
$\longrightarrow$ (deleg K_{phone} (key k) (actionF (file f) wf))

K_{laptop} signed $\forall$ k a v f (deleg K_{laptop} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{laptop} (key k) (actionRMD (attr K_{laptop} a v) :: nil (file f) rmd))

$K_{heather}$ signed $\forall$ k a v f (deleg $K_{heather}$ (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{heather}$ (key k) (actionRMD (attr K_{laptop} a v) :: nil (file f) rmd))

$K_{heather}$ signed $\forall$ k a v f o (deleg $K_{heather}$ $K_{o.k}$ (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{heather}$ $K_{o.k}$ (actionRMD (attr K_{laptop} a v) :: nil (file f) rmd))

K_{laptop} signed deleg K_{laptop} $K_{heather}$ (actionD K_{laptop} cf)

K_{laptop} signed deleg K_{laptop} $K_{heather}$ (actionD K_{laptop} cmd)

K_{laptop} signed $\forall$ *q1* k f (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{laptop} (key k) (actionRMD q1 (file f) rmd))

K_{laptop} signed $\forall$ *q1* k f (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{laptop} (key k) (actionRMD q1 (file f) dmd))

K_{laptop} signed $\forall$ f k a v (attrForm K_{laptop} "creator" k (file f))
$\longrightarrow$ (deleg K_{laptop} (key k) (actionRMD (attr K_{laptop} a v) :: nil (file f) rmd))

K_{laptop} signed $\forall$ f k (attrForm K_{laptop} "creator" k (file f))
$\longrightarrow$ (deleg K_{laptop} (key k) (actionF (file f) rf))

K_{laptop} signed $\forall$ f k (attrForm K_{laptop} "creator" k (file f))
$\longrightarrow$ (deleg K_{laptop} (key k) (actionF (file f) wf))

K_{dvr} signed $\forall$ k a v f (deleg K_{dvr} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{dvr} (key k) (actionRMD (attr K_{dvr} a v) :: nil (file f) rmd))

$K_{husband}$ signed $\forall$ k a v f (deleg $K_{husband}$ (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{husband}$ (key k) (actionRMD (attr K_{dvr} a v) :: nil (file f) rmd))

$K_{husband}$ signed $\forall\ k\ a\ v\ f\ o$ (deleg $K_{husband}\ K_{o.k}$ (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{husband}\ K_{o.k}$ (actionRMD (attr $K_{dvr}\ a\ v$) :: nil (file f) rmd))

K_{dvr} signed deleg $K_{dvr}\ K_{husband}$ (actionD K_{dvr} cf)

K_{dvr} signed deleg $K_{dvr}\ K_{husband}$ (actionD K_{dvr} cmd)

K_{dvr} signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{dvr} (key k) (actionRMD q1 (file f) rmd))

K_{dvr} signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{dvr} (key k) (actionRMD q1 (file f) dmd))

K_{dvr} signed $\forall\ f\ k\ a\ v$ (attrForm K_{dvr} "creator" k (file f))
$\longrightarrow$ (deleg K_{dvr} (key k) (actionRMD (attr $K_{dvr}\ a\ v$) :: nil (file f) rmd))

K_{dvr} signed $\forall\ f\ k$ (attrForm K_{dvr} "creator" k (file f))
$\longrightarrow$ (deleg K_{dvr} (key k) (actionF (file f) rf))

K_{dvr} signed $\forall\ f\ k$ (attrForm K_{dvr} "creator" k (file f))
$\longrightarrow$ (deleg K_{dvr} (key k) (actionF (file f) wf))

K_{camera} signed $\forall\ k\ a\ v\ f$ (deleg K_{camera} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{camera} (key k) (actionRMD (attr $K_{camera}\ a\ v$) :: nil (file f) rmd))

$K_{heather}$ signed $\forall\ k\ a\ v\ f$ (deleg $K_{heather}$ (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{heather}$ (key k) (actionRMD (attr $K_{camera}\ a\ v$) :: nil (file f) rmd))

$K_{heather}$ signed $\forall\ k\ a\ v\ f\ o$ (deleg $K_{heather}\ K_{o.k}$ (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{heather}\ K_{o.k}$ (actionRMD (attr $K_{camera}\ a\ v$) :: nil (file f) rmd))

K_{camera} signed deleg $K_{camera}\ K_{heather}$ (actionD K_{camera} cf)

K_{camera} signed deleg $K_{camera}\ K_{heather}$ (actionD K_{camera} cmd)

K_{camera} signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{camera} (key k) (actionRMD q1 (file f) rmd))

K_{camera} signed $\forall\ q1\ k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{camera} (key k) (actionRMD q1 (file f) dmd))

K_{camera} signed $\forall\ f\ k\ a\ v$ (attrForm K_{camera} "creator" k (file f))
$\longrightarrow$ (deleg K_{camera} (key k) (actionRMD (attr $K_{camera}\ a\ v$) :: nil (file f) rmd))

K_{camera} signed $\forall\ f\ k$ (attrForm K_{camera} "creator" k (file f))
$\longrightarrow$ (deleg K_{camera} (key k) (actionF (file f) rf))

K_{camera} signed $\forall\ f\ k$ (attrForm K_{camera} "creator" k (file f))
$\longrightarrow$ (deleg K_{camera} (key k) (actionF (file f) wf))

K_{ipod} signed $\forall\ k\ a\ v\ f$ (deleg K_{ipod} (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg K_{ipod} (key k) (actionRMD (attr $K_{ipod}\ a\ v$) :: nil (file f) rmd))

$K_{heather}$ signed $\forall\ k\ a\ v\ f$ (deleg $K_{heather}$ (key k) (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{heather}$ (key k) (actionRMD (attr $K_{ipod}\ a\ v$) :: nil (file f) rmd))

$K_{heather}$ signed $\forall\ k\ a\ v\ f\ o$ (deleg $K_{heather}\ K_{o.k}$ (actionF (file f) sf))
$\longrightarrow$ (deleg $K_{heather}\ K_{o.k}$ (actionRMD (attr $K_{ipod}\ a\ v$) :: nil (file f) rmd))

K_{ipod} signed deleg $K_{ipod}\ K_{heather}$ (actionD K_{ipod} cf)

K_{ipod} signed deleg $K_{ipod}\ K_{heather}$ (actionD K_{ipod} cmd)

K_{ipod} signed $\forall$ *q1* $k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{ipod} (key k) (actionRMD q1 (file f) rmd))

K_{ipod} signed $\forall$ *q1* $k\ f$ (isOwner (key k) q1)
$\longrightarrow$ (deleg K_{ipod} (key k) (actionRMD q1 (file f) dmd))

K_{ipod} signed $\forall\ f\ k\ a\ v$ (attrForm K_{ipod} "creator" k (file f))
$\longrightarrow$ (deleg K_{ipod} (key k) (actionRMD (attr $K_{ipod}\ a\ v$) :: nil (file f) rmd))

K_{ipod} signed $\forall\ f\ k$ (attrForm K_{ipod} "creator" k (file f))
$\longrightarrow$ (deleg K_{ipod} (key k) (actionF (file f) rf))

K_{ipod} signed $\forall\ f\ k$ (attrForm K_{ipod} "creator" k (file f))
$\longrightarrow$ (deleg K_{ipod} (key k) (actionF (file f) wf))

K_{phone} signed $\forall\ f$ (deleg $K_{phone}\ K_{heather}$ (actionF (file f) sf))

K_{phone} signed $\forall\ f$ (deleg $K_{phone}\ K_{heather}$ (actionF (file f) rf))

K_{phone} signed $\forall\ f$ (deleg $K_{phone}\ K_{heather}$ (actionF (file f) wf))

K_{phone} signed $\forall\ q\ f$ (deleg $K_{phone}\ K_{heather}$ (actionRMD q (file f) rmd))

K_{phone} signed $\forall\ q\ f$ (deleg $K_{phone}\ K_{heather}$ (actionRMD q (file f) dmd))

K_{laptop} signed $\forall\ f$ (deleg $K_{laptop}\ K_{heather}$ (actionF (file f) sf))

K_{laptop} signed $\forall\ f$ (deleg $K_{laptop}\ K_{heather}$ (actionF (file f) rf))

K_{laptop} signed $\forall\, f$ (deleg K_{laptop} $K_{heather}$ (actionF (file f) wf))

K_{laptop} signed $\forall\, q\ f$ (deleg K_{laptop} $K_{heather}$ (actionRMD q (file f) rmd))

K_{laptop} signed $\forall\, q\ f$ (deleg K_{laptop} $K_{heather}$ (actionRMD q (file f) dmd))

K_{dvr} signed $\forall\, f$ (deleg K_{dvr} $K_{husband}$ (actionF (file f) sf))

K_{dvr} signed $\forall\, f$ (deleg K_{dvr} $K_{husband}$ (actionF (file f) rf))

K_{dvr} signed $\forall\, f$ (deleg K_{dvr} $K_{husband}$ (actionF (file f) wf))

K_{dvr} signed $\forall\, q\ f$ (deleg K_{dvr} $K_{husband}$ (actionRMD q (file f) rmd))

K_{dvr} signed $\forall\, q\ f$ (deleg K_{dvr} $K_{husband}$ (actionRMD q (file f) dmd))

K_{camera} signed $\forall\, f$ (deleg K_{camera} $K_{heather}$ (actionF (file f) sf))

K_{camera} signed $\forall\, f$ (deleg K_{camera} $K_{heather}$ (actionF (file f) rf))

K_{camera} signed $\forall\, f$ (deleg K_{camera} $K_{heather}$ (actionF (file f) wf))

K_{camera} signed $\forall\, q\ f$ (deleg K_{camera} $K_{heather}$ (actionRMD q (file f) rmd))

K_{camera} signed $\forall\, q\ f$ (deleg K_{camera} $K_{heather}$ (actionRMD q (file f) dmd))

K_{ipod} signed $\forall\, f$ (deleg K_{ipod} $K_{heather}$ (actionF (file f) sf))

K_{ipod} signed $\forall\, f$ (deleg K_{ipod} $K_{heather}$ (actionF (file f) rf))

K_{ipod} signed $\forall\, f$ (deleg K_{ipod} $K_{heather}$ (actionF (file f) wf))

K_{ipod} signed $\forall\, q\ f$ (deleg K_{ipod} $K_{heather}$ (actionRMD q (file f) rmd))

K_{ipod} signed $\forall\, q\ f$ (deleg K_{ipod} $K_{heather}$ (actionRMD q (file f) dmd))

$K_{heather}$ signed K_{laptop} speaksfor $K_{heather.trustedDevices}$

$K_{heather}$ signed K_{phone} speaksfor $K_{heather.trustedDevices}$

$K_{heather}$ signed K_{ipod} speaksfor $K_{heather.trustedDevices}$

$K_{heather}$ signed K_{camera} speaksfor $K_{heather.trustedDevices}$

$K_{heather}$ signed $K_{heather.trustedDevices}$ speaksfor $K_{heather}$

$K_{husband}$ signed K_{dvr} speaksfor $K_{husband.trustedDevices}$

$K_{husband}$ signed $K_{husband.trustedDevices}$ speaksfor $K_{husband}$

$K_{heather}$ signed $K_{coworker1}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker10}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker11}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker12}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker13}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker14}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker2}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker3}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker4}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker5}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker6}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker7}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker8}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{coworker9}$ speaksfor $K_{heather.coworkers}$

$K_{heather}$ signed $K_{friend1}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend10}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend11}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend12}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend13}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend14}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend2}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend3}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend4}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend5}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend6}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend7}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend8}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{friend9}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed K_{guest1} speaksfor $K_{heather.guests}$

$K_{heather}$ signed $K_{guest10}$ speaksfor $K_{heather.guests}$

$K_{heather}$ signed $K_{guest11}$ speaksfor $K_{heather.guests}$

$K_{heather}$ signed $K_{guest12}$ speaksfor $K_{heather.guests}$

$K_{heather}$ signed $K_{guest13}$ speaksfor $K_{heather.guests}$

$K_{heather}$ signed $K_{guest14}$ speaksfor $K_{heather.guests}$

$K_{heather}$ signed K_{guest2} speaksfor $K_{heather.guests}$

$K_{heather}$ signed K_{guest3} speaksfor $K_{heather.guests}$

$K_{heather}$ signed K_{guest4} speaksfor $K_{heather.guests}$

$K_{heather}$ signed K_{guest5} speaksfor $K_{heather.guests}$

$K_{heather}$ signed K_{guest6} speaksfor $K_{heather.guests}$

$K_{heather}$ signed K_{guest7} speaksfor $K_{heather.guests}$

$K_{heather}$ signed K_{guest8} speaksfor $K_{heather.guests}$

$K_{heather}$ signed K_{guest9} speaksfor $K_{heather.guests}$

$K_{heather}$ signed $K_{relative1}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative10}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative11}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative12}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative13}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative14}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative2}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative3}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative4}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative5}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative6}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative7}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative8}$ speaksfor $K_{heather.friends}$

$K_{heather}$ signed $K_{relative9}$ speaksfor $K_{heather.friends}$

$K_{husband}$ signed K_{guest1} speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed $K_{guest10}$ speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed $K_{guest11}$ speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed $K_{guest12}$ speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed $K_{guest13}$ speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed $K_{guest14}$ speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed K_{guest2} speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed K_{guest3} speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed K_{guest4} speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed K_{guest5} speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed K_{guest6} speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed K_{guest7} speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed K_{guest8} speaksfor $K_{husband.tvguests}$

$K_{husband}$ signed K_{guest9} speaksfor $K_{husband.tvguests}$

$K_{heather}$ signed K_{boss} speaksfor $K_{heather.coworkers}$

K_{laptop} signed deleg K_{laptop} $K_{husband}$ (actionD K_{laptop} cf)

K_{laptop} signed deleg K_{laptop} $K_{husband}$ (actionD K_{laptop} cmd)

$K_{heather}$ signed $\forall$ f (deleg $K_{heather}$ $K_{husband}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall$ q f (deleg $K_{heather}$ $K_{husband}$ (actionRMD q (file f) rmd))

$K_{husband}$ signed $\forall$ f (deleg $K_{husband}$ $K_{heather}$ (actionF (file f) rf))

$K_{husband}$ signed $\forall$ q f (deleg $K_{husband}$ $K_{heather}$ (actionRMD q (file f) rmd))

$K_{heather}$ signed $\forall$ f (deleg $K_{heather}$ $K_{husband}$ (actionF (file f) wf))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.household}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall$ f (deleg $K_{heather}$ $K_{heather.household}$ (actionRMD (attr $K_{heather}$ "type" "photo") :: nil (file f) rmd))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.household}$ (actionF (file f) sf))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.coworkers}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall$ f (deleg $K_{heather}$ $K_{heather.coworkers}$ (actionRMD (attr $K_{heather}$ "type" "photo") :: nil (file f) rmd))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.coworkers}$ (actionF (file f) sf))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "music" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.coworkers}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall$ f (deleg $K_{heather}$ $K_{heather.coworkers}$ (actionRMD (attr $K_{heather}$ "type" "music") :: nil (file f) rmd))

$K_{heather}$ signed $\forall f$ (attrForm $K_{heather}$ "type" "music" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.coworkers}$ (actionF (file f) sf))

$K_{heather}$ signed $\forall f$ (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{daughter}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall f$ (deleg $K_{heather}$ $K_{daughter}$ (actionRMD (attr $K_{heather}$ "type" "photo") :: nil (file f) rmd))

$K_{heather}$ signed $\forall f$ (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{daughter}$ (actionF (file f) sf))

$K_{heather}$ signed $\forall f$ (attrForm $K_{heather}$ "type" "music" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{daughter}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall f$ (deleg $K_{heather}$ $K_{daughter}$ (actionRMD (attr $K_{heather}$ "type" "music") :: nil (file f) rmd))

$K_{heather}$ signed $\forall f$ (attrForm $K_{heather}$ "type" "music" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{daughter}$ (actionF (file f) sf))

$K_{husband}$ signed $\forall f$ (attrForm $K_{husband}$ "type" "tvshow" (file f)) $\wedge$ (attrForm $K_{husband}$ "inappropriate" "No" (file f))
$\longrightarrow$ (deleg $K_{husband}$ $K_{daughter}$ (actionF (file f) rf))

$K_{husband}$ signed $\forall f$ (deleg $K_{husband}$ $K_{daughter}$ (actionRMD (attr $K_{husband}$ "inappropriate" "No") :: (attr $K_{husband}$ "type" "tvshow") :: nil (file f) rmd))

$K_{husband}$ signed $\forall f$ (attrForm $K_{husband}$ "type" "tvshow" (file f)) $\wedge$ (attrForm $K_{husband}$ "inappropriate" "No" (file f))
$\longrightarrow$ (deleg $K_{husband}$ $K_{daughter}$ (actionF (file f) sf))

$K_{heather}$ signed $\forall f$ (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.friends}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall f$ (deleg $K_{heather}$ $K_{heather.friends}$ (actionRMD (attr $K_{heather}$ "type" "photo") :: nil (file f) rmd))

$K_{heather}$ signed $\forall f$ (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.friends}$ (actionF (file f) sf))

$K_{husband}$ signed $\forall f$ (attrForm $K_{husband}$ "type" "tvshow" (file f))
$\longrightarrow$ (deleg $K_{husband}$ $K_{heather.friends}$ (actionF (file f) rf))

$K_{husband}$ signed $\forall f$ (deleg $K_{husband}$ $K_{heather.friends}$ (actionRMD (attr $K_{husband}$ "type" "tvshow") :: nil (file f) rmd))

$K_{husband}$ signed $\forall f$ (attrForm $K_{husband}$ "type" "tvshow" (file f))

$\longrightarrow$ (deleg $K_{husband}$ $K_{heather.friends}$ (actionF (file f) sf))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "music" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.friends}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall$ f (deleg $K_{heather}$ $K_{heather.friends}$ (actionRMD (attr $K_{heather}$ "type" "music") :: nil (file f) rmd))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "music" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.friends}$ (actionF (file f) sf))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.guests}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall$ f (deleg $K_{heather}$ $K_{heather.guests}$ (actionRMD (attr $K_{heather}$ "type" "photo") :: nil (file f) rmd))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "photo" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.guests}$ (actionF (file f) sf))

$K_{husband}$ signed $\forall$ f (attrForm $K_{husband}$ "type" "tvshow" (file f))
$\longrightarrow$ (deleg $K_{husband}$ $K_{husband.tvguests}$ (actionF (file f) rf))

$K_{husband}$ signed $\forall$ f (deleg $K_{husband}$ $K_{husband.tvguests}$ (actionRMD (attr $K_{husband}$ "type" "tvshow") :: nil (file f) rmd))

$K_{husband}$ signed $\forall$ f (attrForm $K_{husband}$ "type" "tvshow" (file f))
$\longrightarrow$ (deleg $K_{husband}$ $K_{husband.tvguests}$ (actionF (file f) sf))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "music" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.guests}$ (actionF (file f) rf))

$K_{heather}$ signed $\forall$ f (deleg $K_{heather}$ $K_{heather.guests}$ (actionRMD (attr $K_{heather}$ "type" "music") :: nil (file f) rmd))

$K_{heather}$ signed $\forall$ f (attrForm $K_{heather}$ "type" "music" (file f))
$\longrightarrow$ (deleg $K_{heather}$ $K_{heather.guests}$ (actionF (file f) sf))