Evaluating the Usability of Usage Controls in Electronic Collaboration

José C. Brustoloni, Ricardo Villamarín-Salomón, Peter Djalaliev and David Kyle
Dept. of Computer Science, University of Pittsburgh
210 S. Bouquet St. #6111, Pittsburgh, PA 15260, USA
{jcb,rvillsal,peterdj,dkyle}@cs.pitt.edu

ABSTRACT

Currently, collaborations often require non-disclosure agreements (NDAs). NDAs can be time-consuming and expensive to negotiate and enforce. Usage controls could be an atractive alternative or adjunct to NDAs. Usage controls enable the distributor of a file to limit how recipients of that file may use it. However, existing usage controls (e.g., PDF's) often are software-based and easy to break. They may not interoperate, and their impact on collaborative workflows is typically unknown. We designed and implemented operating system and Web server and browser modifications that allow hardware-based usage controls to be easily added to existing software-based ones. This paper describes and evaluates our system's user interfaces. In a user study, untrained users role-played design engineers in two similar collaborative scenarios with or without usage controls. Users found the interfaces easy to use, and usage controls had insignificant impact on the completion times and accuracy of the assigned tasks. These results suggest that our usage control approach can add security to collaborative workflows with minimal training and performance penalties.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection—*access controls, authentication, cryptographic controls*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*client/server, distributed applications*; H.5.2 [**Information Interfaces and Presentation (e.g., HCI)**]: User Interfaces—*graphical user interfaces (GUI), evaluation*

General Terms

Security, Human Factors

Keywords

Usage controls, digital rights management, electronic collaboration, Trusted Platform Module (TPM)

1. INTRODUCTION

Organizations are increasingly turning to collaboration in order to remain competitive. More and more manufacturers are collaborating in new designs to reduce costs and time-to-market. Similarly, many retailers are providing detailed forecasts to suppliers to avoid stock-outs while minimizing inventory.

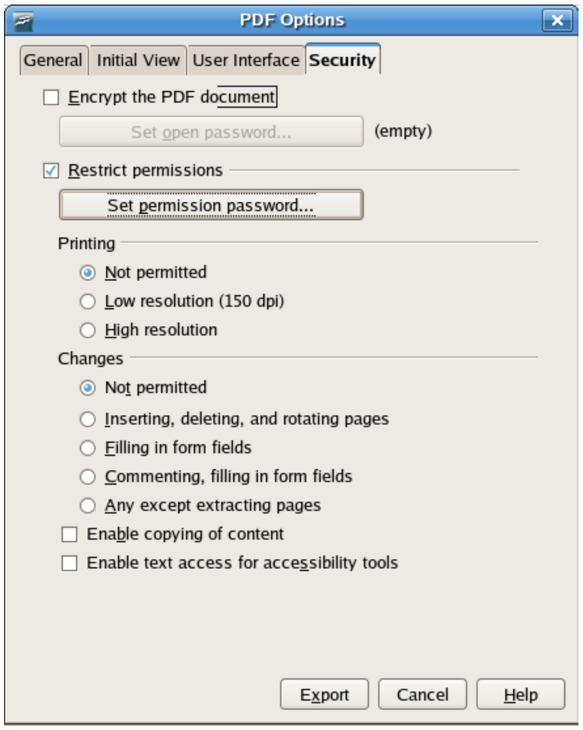
While collaboration can make economic sense, it can also carry great risks. When a manufacturer shares product or process details, it also risks its intellectual property. Likewise, disclosure of a retailer's forecasts to competitors can be put the retailer at disadvantage.

Organizations can, to some extent, use non-disclosure agreements (NDAs) to mitigate such risks. However, NDAs can be time-consuming and expensive to negotiate and enforce. More efficient alternatives are needed.

In response to this need, computer applications increasingly provide usage controls. Usage control enables the distributor of a file to limit how recipients of that file may use it. For example, recent versions of Adobe Acrobat and OpenOffice [1] allow the creator of a PDF document to set flags that prohibit printing or copying content from the document [2].

However, existing usage controls typically have several shortcomings. First, most of them are purely software-based and can be defeated by tools or techniques that can be easily found on the Web. For example, instructions for defeating PDF's usage controls are readily available [3]. Second, there is no standard or uniform method for usage control. Therefore, usage controls of different applications or systems often do not interoperate. Third, little is known about the usability of usage control in collaborations. If usage controls disrupt collaborative workflows, they would negate the benefits collaboration was supposed to give.

To overcome the first two of these shortcomings, we designed and implemented operating system modifications that add hardware-based usage controls to existing software-based ones. No modification is needed in applications for hardware-based protection; the modified operating system needs only the secure digests of the trusted applications. Hardware-based usage policies are enforced by the operating system (e.g., UCLinux [4]) in conjunction with a secure coprocessor (Trusted Platform Module (TPM) [5]). Usage-controlled files are stored in an encrypted file system whose secret key the TPM reveals only when the system is in a trustworthy state. Usage-controlled files may contain software-based usage policies that applications that open the file are expected to enforce (e.g., PDF print restrictions). We add to such files, as metadata, hardware-based usage policies.



☐ Encrypt the PDF document

Set open password... (empty)

☑ Restrict permissions

Set permission password...

Printing

[✓ Not permitted] Low resolution (150 dpi) High resolution

Changes

[✓ Not permitted] Inserting, deleting, and rotating pages Filling in form fields Commenting, filling in form fields Any except extracting pages

☐ Enable copying of content

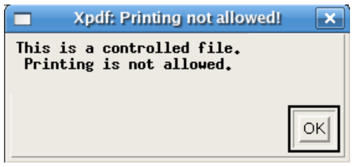
☐ Enable text access for accessibility tools

Figure 1: OpenOffice dialog for setting security options in PDF files

These policies may, e.g., restrict the time period and applications that can open a file. By restricting the applications that can open a file, hardware-based usage policies prevent circumvention of the software-based usage policies contained in the file. For example, only trusted versions of Acrobat Reader could be able to open usage-controlled PDF files, while cracked versions of xpdf [6] would be unable to do so. We also designed and implemented Web server and browser modifications that enable secure transmission of usage-controlled files. The modified Web server sends such a file to a client only if the client provides an *attestation* that the client's platform is trustworthy, i.e., will enforce the usage policies received from the server. This attestation is signed by the client's TPM.

This paper describes and evaluates the user interfaces we designed for authoring and accepting usage policies in our system, overcoming the third above-mentioned shortcoming of existing usage controls. In a user study, untrained users role-played design engineers in two similar collaborative scenarios with or without usage controls. We found that usage controls had insignificant impact on the completion times and accuracy of the assigned tasks. Users found the interfaces easy to use. They had insignificant trouble in properly considering documents' usage policies when deciding whether to accept them and in generating documents with appropriate usage policies. These results suggest that our usage control approach can add security to collaborative workflows with minimal training and performance penalties.

The rest of this paper is organized as follows. Section 2 summarizes relevant aspects of software used in this paper. Section 3 describes our user interfaces, and Section 4 evaluates them in a user study. Finally, Section 5 discusses related work, and Section 6 concludes.



```
This is a controlled file.
 Printing is not allowed.
```

Figure 2: xpdf dialog for unprintable files

2. BACKGROUND

This section provides a brief overview of relevant features of software used in this paper.

2.1 Authoring and viewing software-based usage policies

Fig. 1 displays OpenOffice's dialog for setting security options in PDF files. Such a dialog has been available since OpenOffice v. 2.0 [1]. A PDF document's author can disable high-resolution or any printing, extracting pages, commenting, filling in form fields, inserting, deleting, and rotating pages, or any changes, content copying, and text access for accessibility tools.

The xpdf viewer [6] disables certain options, depending on the security settings of the PDF file being viewed. For example, Fig. 2 shows the dialog box that xpdf displays when the user clicks on the icon for printing a file that has a usage policy that prohibits printing. It should be noted, however, that PDF is an open standard [2], and xpdf if open-source software. Instructions for how to modify xpdf so as to ignore usage policies in PDF files can be easily found on the Web [3]. Weaknesses in software-based usage controls, such as PDF's, motivate our effort to add to them hardware-based usage controls.

2.2 Operating system modifications

We designed and implemented a Linux Security Module [7], UCLinux [4], that adds support for TPM-based usage controls in the Linux operating system. UCLinux requires that the computer have a TPM v. 1.1b [5] or later and TPM-aware BIOS and boot loader (e.g., GRUB). Computers with TPM and TPM-aware BIOS are commercially available from Lenovo, Dell, HP, and other manufacturers. During boot, the BIOS measures the integrity of the boot loader and extends the result into a platform configuration register (PCR) in the TPM. (PCR values are erased when the system is reset. When a value V is extended into a PCR, the TPM concatenates V and the PCR's current value, computes the secure digest, and stores the result into the PCR. The only two ways to modify a PCR are to reset the system or extend a value into the PCR.) The boot loader likewise measures the integrity of the kernel and extends the result into a PCR. The kernel then performs *TCB prelogging*: it optimistically extends into a PCR the expected measurements of every component in the system's Trusted Computing Base (TCB). These expected measurements are found in a configuration file called TCB list.

The system's usage-controlled file system (UCFS) has its secret key sealed by the TPM to the PCR values that result from the boot sequence and TCB list. Thus, at boot time, the kernel can retrieve the UCFS's secret key from the

TPM and mount the UCFS. Hardware-based usage policies are stored in the respective file's extended attributes. A file's hardware-based usage policy may specify the integrity measurements of programs allowed to open the file. A hardware-based usage policy may be *prepared* by the respective file's distributor, or *received* by the file's recipient. The kernel enforces only the latter.

During runtime, the kernel measures the integrity measurement of the programs that it executes. When a program opens or memory-maps a file with received hardware-based usage policies, the kernel verifies that access is permitted by those policies. If the current process's integrity measurement differs from those specified in the file's received policies, the open or memory-map call fails.

The kernel also verifies the actual integrity measurement of each TCB component that it executes, memory-maps, or opens. If an actual measurement is different from the one in the TCB list, or a privileged program that is not in the TCB list is about to be executed, or a privileged user attempts to log interactively into the system, the system's integrity and trustworthiness may soon be violated. The kernel then performs an operation called *root trip*: the kernel aborts any processes with a UCFS file open, unmounts the UCFS, and erases any copy in memory of UCFS's secret key and memory freed as a result of processes' termination and UCFS's unmounting. The kernel also records this event into a PCR. The kernel then continues the process that caused the root trip. The system thereafter can be used normally, but UCFS contents will be unavailable. Because the UCFS's key is sealed to the boot-time PCR values, the system will have to be rebooted and brought to a trustworthy state before the UCFS can be mounted again.

2.3 Web server and browser modifications

We also designed Web server and browser modifications for securely transmitting usage-controlled files, and implemented them by modifying Apache and Firefox, respectively. The modified browser depends on the modified operating system (e.g., UCLinux), but the modified server does not.

When a file with prepared hardware-based usage policies is requested, the modified server returns to the client these policies and asks the client to upgrade the connection to TLS with a TLS extension. The modified browser displays the policies to the user. If the user accepts them, the browser initiates the connection upgrade. Using the TLS extension, the server obtains an attestation from the client. The attestation includes a *quote*, i.e., the server's nonce and the client's PCR values signed by the client's TPM, an attestation identity certificate (AIC), and a measurement log containing the name and value of every integrity measurement extended into the PCRs. The server uses its nonce and the AIC to verify the quote and uses the quote's PCR values to verify the measurement log. If the server trusts the configuration represented by the measurement log, the server completes the connection upgrade. The browser then gets the usage-controlled file over the upgraded connection, and stores it in the client's UCFS.

The browser is part of the client's TCB list and its integrity measurement is revealed to the server during the client's attestation. To prevent abuses, the client's operating system ensures that only this browser can obtain quotes and write received hardware-based usage policies. Any program can prepare hardware-based usage policies, but prepared policies are not enforced locally.

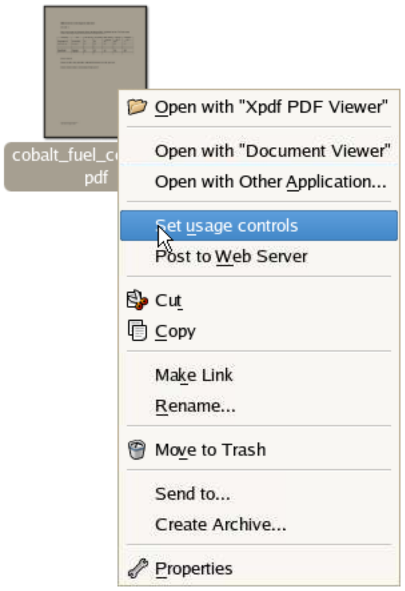


Figure 3: Contextual menu option for setting a file's hardware-based usage policies

3. USER INTERFACE DESIGN

This section describes our user interfaces for authoring and viewing usage policies.

3.1 Plug-ins for uniformly viewing software-based usage policies

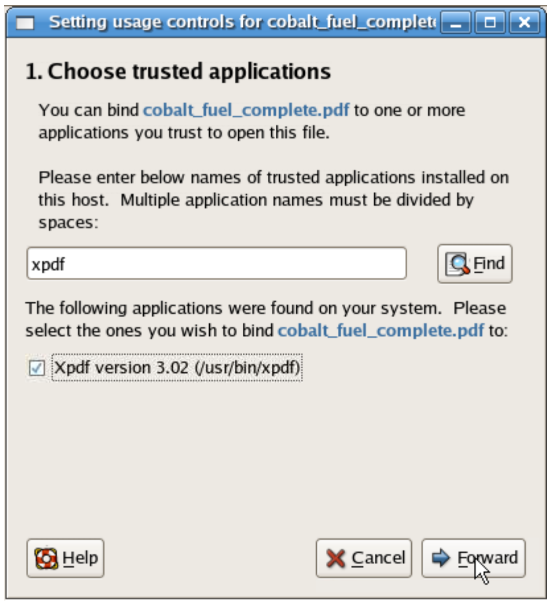
To provide a consistent user experience, it is desirable that user-interface programs display hardware- and software-based usage policies in an integrated fashion. However, software-based usage policies may be represented in various ways by different applications. We enable consistent and integrated display of usage policies by configuring the system with policy translator plug-ins for each supported file extension (e.g., .pdf). A policy translator translates a file's software-based usage policies to a common format. In our implementation, the common format is an extension of the Open Rigital Rights Language (ODRL) [8]. The user-interface programs, operating system, and hardware-based usage controls use this format natively.

Policy translators are optional. They are used only by user-interface programs for displaying software-based usage controls. A file can have hardware-based usage policies and, if received, our system will enforce them even if the file's extension does not have a configured policy translator.

3.2 Authoring hardware-based usage policies

We added to the operating system's file manager a context-sensitive command for setting a file's hardware-based usage policies. The user accesses the command by right-clicking the file and selecting from a menu, as shown in Fig. 3. We implemented this command in Nautilus, the file manager that is part of the Gnome desktop environment used in many Linux systems.

The command's first dialog box is shown in Fig. 4. The user enters the name of a program that is trusted to open the


Setting usage controls for cobalt_fuel_complet

1. Choose trusted applications

You can bind cobalt_fuel_complete.pdf to one or more applications you trust to open this file.

Please enter below names of trusted applications installed on this host. Multiple application names must be divided by spaces:

xpdf

The following applications were found on your system. Please select the ones you wish to bind cobalt_fuel_complete.pdf to:

☑ Xpdf version 3.02 (/usr/bin/xpdf)

Figure 4: Binding file to trusted application

file. The command searches for programs with that name in the directories in the user's path environment variable. If more than one instance is found, the user clicks on the desired instances. The command then transparently creates a usage policy with the name and integrity measurements of the program versions trusted to open the file.

The second dialog box is displayed in Fig. 5. The user selects the starting and ending dates of the period during which access to the file is allowed. The command transparently creates a usage policy with this period. The dialog also displays the file's software-based usage policies, if the file has an extension for which a policy translator is configured (e.g., .pdf). When the user clicks OK, the command links the prepared usage policies to the file.

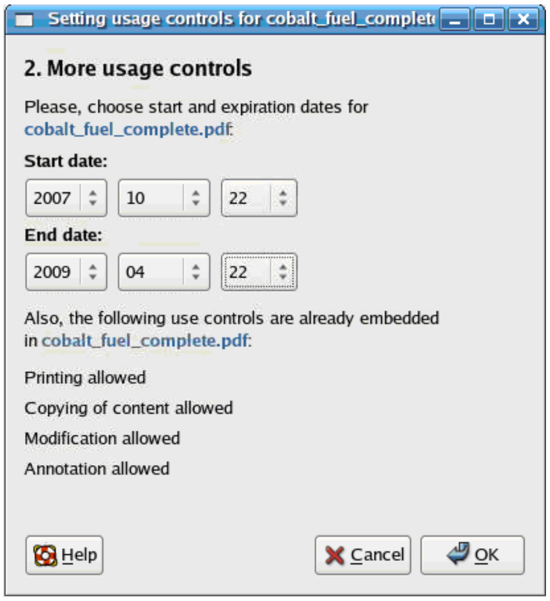
A file's author or distributor does not need a modified operating system or TPM. Even if the author or distributor has them, the operating system enforces only received usage policies, and only the modified browser can write such policies. Consequently, usage policies are enforced only at clients. The distributor does need, however, a modified Web server that verifies that clients are trustworthy (as attested by a TPM) before sending usage-controlled files to them.

3.3 Overriding usage policies

System administrators can configure in each user's computer usage policies that override the user's policies or decisions when the user attempts to post or retrieve a usage-controlled file. In our current implementation, those overriding policies are written using a conventional text editor, directly in our ODRL extension. For example, a overriding policy may specify that posted files must not allow printing or copying, or that acceptable files must be accessible for at least a year.

3.4 Posting usage-controlled files to Web site

To facilitate posting usage-controlled files to a Web site, we added to the operating system's file manager another context-sensitive command. As shown in Fig. 6, the user


Setting usage controls for cobalt_fuel_complet

2. More usage controls

Please, choose start and expiration dates for cobalt_fuel_complete.pdf:

Start date: 2007 10 22

End date: 2009 04 22

Also, the following use controls are already embedded in cobalt_fuel_complete.pdf:

Printing allowed

Copying of content allowed

Modification allowed

Annotation allowed

Figure 5: Specifying allowed period for accessing file

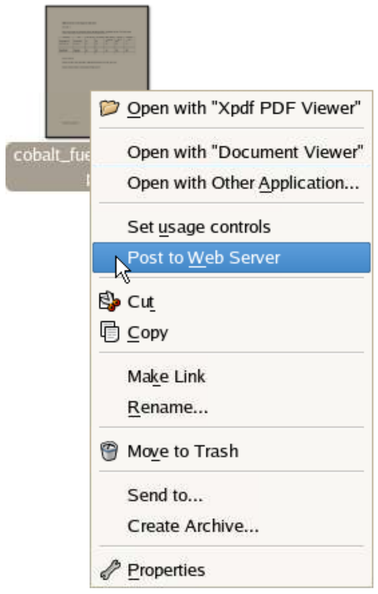
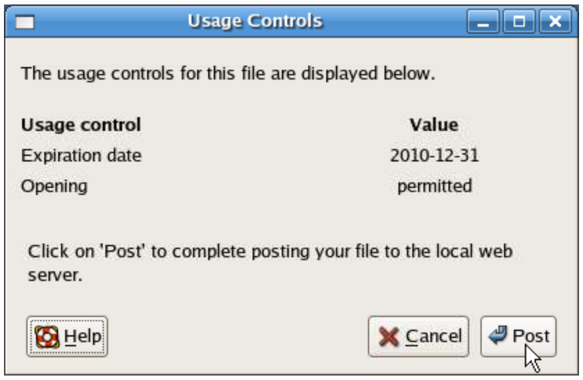


Figure 6: Contextual menu option for posting a file to Web site

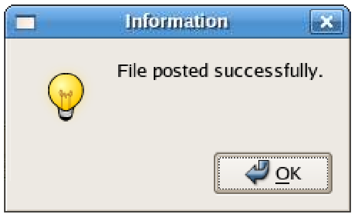


The usage controls for this file are displayed below.

Usage control	Value
Expiration date	2010-12-31
Opening	permitted

Click on 'Post' to complete posting your file to the local web server.

Figure 7: Dialog for confirming or canceling posting of file (without overriding policies)



File posted successfully.

Figure 8: Confirmation of file posting

accesses the command by right-clicking the file.

If the system is configured without overriding policies for posting files, the command simply displays the file's usage policies and asks the user to confirm or cancel the operation, as shown in Fig. 7. A configuration file specifies to what subdirectory in the Web server's directory the command should copy the posted file. If the user confirms, the command displays the box shown in Fig. 8.

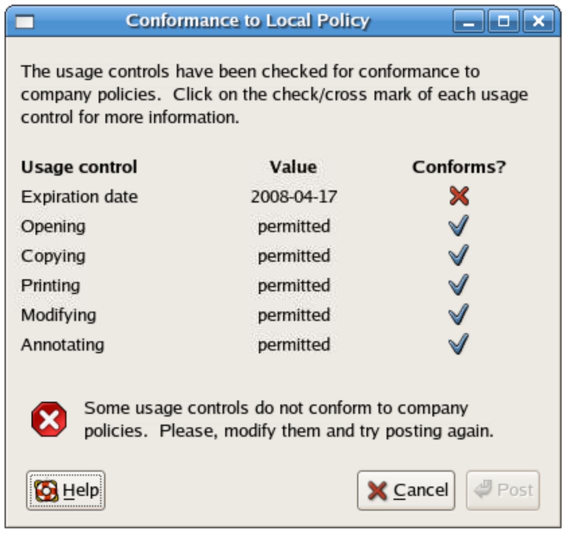
On the contrary, if there are overriding policies for posting files, the command checks whether the file is compliant to those policies and presents the results in the dialog box shown in Fig. 9. If the file is noncompliant, the option for confirming the operation is grayed out; the user can click on the icons for more information, or cancel.

3.5 Retrieving usage-controlled files from Web site

Before the client's browser downloads a usage-controlled file, the browser gets the file's usage policies from the server. If the system is configured without overriding policies for accepting files, the browser simply displays the file's usage policies and asks the user to confirm or cancel the operation, as shown in Fig. 10.

On the contrary, if there are overriding policies for accepting files, the browser checks whether the file complies with those policies. If the file is noncompliant, the browser displays the dialog shown in Fig. 11. The user can click on the icons for more information or cancel.

The browser stores usage-controlled files in the UCFS, whose key is managed by the TPM. There is a subdirectory in this file system for each user, e.g., /controlled/user/pat/. The browser does not allow a user to store a downloaded usage-controlled file outside his or her subdirectory in the UCFS.

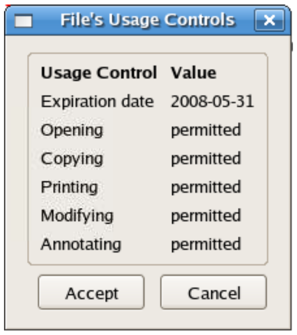


The usage controls have been checked for conformance to company policies. Click on the check/cross mark of each usage control for more information.

Usage control	Value	Conforms?
Expiration date	2008-04-17	✗
Opening	permitted	✓
Copying	permitted	✓
Printing	permitted	✓
Modifying	permitted	✓
Annotating	permitted	✓

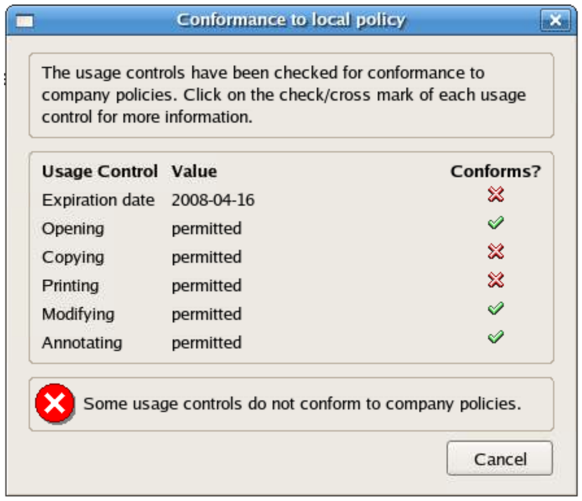
Some usage controls do not conform to company policies. Please, modify them and try posting again.

Figure 9: Dialog for confirming or canceling posting of file (with overriding policies)



Usage Control	Value
Expiration date	2008-05-31
Opening	permitted
Copying	permitted
Printing	permitted
Modifying	permitted
Annotating	permitted

Figure 10: Dialog for accepting or canceling file download (without overriding policies)



The usage controls have been checked for conformance to company policies. Click on the check/cross mark of each usage control for more information.

Usage Control	Value	Conforms?
Expiration date	2008-04-16	✗
Opening	permitted	✓
Copying	permitted	✗
Printing	permitted	✗
Modifying	permitted	✓
Annotating	permitted	✓

Some usage controls do not conform to company policies.

Figure 11: Dialog denying file download (with overriding policies)

The client's operating system ensures that any file with received hardware-based usage policies can be opened only by applications that the file's author trusts, as specified in those policies. Usually, trusted applications do not include generic commands such as cp, lpr, or mv. Thus, operations such as copying or printing usage-controlled files can be performed only by specific trusted applications, which enforce software-based usage policies.

4. USABILITY EVALUATION

We performed a user study to evaluate our system and report the results in this section.

4.1 Experimental design

In our user study, each participant role-played a user in a first scenario, and then role-played another user in a similar second scenario. Both scenarios required the user to consider seven documents available for download from the Web, make and write a design decision based on them, and then post the decision to a Web site. To avoid order-induced biases, the first scenario was randomly selected between scenarios F and T (described in the next section), and the remaining scenario was performed second.

Considering that, before the experiment, participants were familiar with Web browsing but unfamiliar with usage controls, the first scenario was always performed without usage controls, using unmodified software. There was one exception, however: the file manager had a contextual menu similar to the one shown in Fig. 6 (without the dialog of Fig. 7), to facilitate posting the decision to the Web site. The second scenario was performed with usage controls, using our modified Web server and client operating system and browser and user interfaces described in Sections 2 and 3. In the second scenario, we asked the user to accept only documents whose usage policies conform to specified policies, and to set certain usage policies in his or her decision.

In each scenario, four of the seven documents (set A) had acceptable usage policies and useful information. The remaining three documents (set B) had unacceptable usage policies and no useful information. We measured under each condition (i.e., with or without usage controls) the number of documents in set A downloaded, the number of documents in set B downloaded, the number of decisions correctly made and posted, and the task completion time. To evaluate the statistical significance of differences between the two conditions, we performed paired t-tests.

Ideally, usage controls would cause significant decrease in the number of documents in set B downloaded, because they have unacceptable usage policies. At the same time, usage controls should cause insignificant difference in the number of documents in set A downloaded, number of decisions correctly made and posted, and task completion time. The overriding policies described in Section 3.3 can guarantee that no documents in set B would be downloaded and no decisions with incorrect usage policies would be posted, regardless of the usability of the user interfaces. To avoid such biases in favor of usage controls, the scenarios did not have any overriding policies.

When a user performs a scenario first, one would expect the user to download all of the scenario's documents, because usage policies are not visible to the user. Conversely, when a user performs the same scenario second, one would expect the user to reject documents in set B, because they have unacceptable policies. If documents in set B contained useful information, their consideration would tend to increase task completion time and might change the decision when the scenario is performed first. To avoid such biases in favor of usage controls, documents in set B contained no useful information.

Table 1: Participant characteristics

# Participants	10
# Male	3
# Female	7
# With Engineering / CS background	5
Ease of understanding user study tasks (SR)	3.8 / 5
Ease of completing user study tasks (SR)	3.6 / 5

4.2 Scenarios

In both scenarios, the participant role-plays an engineer employed by a car manufacturer. In each scenario, the engineer's task is to consider alternatives offered by seven suppliers and, according to specified criteria, select one of them for an upcoming car model. A specified Web page contains links to the suppliers' offerings. Technical details of each offering are provided in a PDF flyer. In scenario F, the alternatives are engines that use alternative fuels, while in scenario T, the alternatives are tires. The flyers of the third, fourth, and seventh suppliers (set B) have unacceptable usage policies and simply say that technical information is still unavailable. The flyers of the remaining suppliers (set A) have acceptable usage policies and contain the technical information necessary for the design decision.

The engineer uses OpenOffice to write his or her decision and export it to PDF. In the condition with usage controls, the engineer uses the command described in Section 3.2 to set usage policies. The engineer then uses the command described in Section 3.4 to post the decision to the Web site.

4.3 Participants

We recruited participants who were at least 19 years old, proficient in English, and familiar with at least one Web browser, PDF viewer, and word processor. We advertised the user study by distributing flyers aroung the University of Pittsburgh's campus and posting for volunteers in Carnegie Mellon University's Center for Behavioral Decision Research website, pittsburgh.craigslist.org, and pittsburgh.backpage.com.

Given that participants would be role-playing engineers, recruitment advertisements sought to attract participants with some education and work experience in Engineering or Computer Science. However, only half of the participants had such background. Table 1 summarizes participant characteristics (where "SR" denotes self-reported). There were 10 participants, 7 of which were female. On average, participants found the user study tasks moderately easy to understand and complete.

4.4 Laboratory sessions

We scheduled individual laboratory sessions for each participant. Each participant's session lasted between 24 and 48 minutes. Participants received between $15 and $22 for their time.

Table 2: Results of user study (paired t-test, n=10)

	mean	std. dev.	eff. size	p-value
# documents with acceptable policies downloaded (set A)				
Without usage controls	4.0	0.0		
With usage controls	4.0	0.0		
Difference	0.0	0.0		not signif.
# documents with unacceptable policies downloaded (set B)				
Without usage controls	3.0	0.0		
With usage controls	0.1	0.316		
Difference	-2.9	0.316	9.2	< 0.001
# decisions correctly posted				
Without usage controls	1.0	0.0		
With usage controls	0.9	0.316		
Difference	-0.1	0.316		not signif.
Task completion time (minutes)				
Without usage controls	17.77	4.51		
With usage controls	18.89	3.95		
Difference	1.09	3.89		not signif.

Table 3: Participant perceptions of usage control system

Would feel comfortable using system again	4.3 / 5
Would recommend system to a friend	3.5 / 5

During a session, we took notes and recorded the participant's computer screen, face, and voice. The recordings helped us debug the scenarios and user interfaces before the user study, and thereafter helped us confirm counts and task completion times. We did not record participant names or other personal information. We report only aggregate results.

4.5 Results

Table 2 show the main results of the user study. The noted effect sizes are Cohen's d; values of 0.2, 0.5, and 0.8 are indicative of small, medium, and large effects, respectively [9].

The results obtained were close to ideal. First, usage controls had no effect in the number of documents downloaded in set A, which had acceptable policies. Second, usage controls strongly reduced the number of documents downloaded in set B, which had unacceptable policies. Only one of ten participants downloaded such a document, but did not open it. After downloading, the participant reread the instructions, again clicked on the link for downloading the document, read the respective usage policies, and canceled. Third, nine out of ten participants successfully posted the correct decision with correct usage policies. One participant wrote the correct decision and posted it successfully, but gave the decision an incorrect usage policy. Fourth, task completion time was slightly higher with usage controls, but the difference was statistically insignificant.

At the end of each laboratory session, we surveyed user perceptions about our usage control system. The results are summarized in Table 3. Participants indicated that they'd be comfortable using the system again and would give friends a moderately positive recommendation.

5. RELATED WORK

Usage controls for applications such as those discussed in this paper are also known as enterprise digital rights management (EDRM), while usage controls for digital mass media are better known as digital rights management (DRM). Usability tends to be more of a concern in EDRM because EDRM may require many users to author and accept usage policies. On the contrary, in DRM, usage policies usually are authored by few specialists and enforced without explicit client acceptance.

Other major differences between DRM and EDRM include the respective legal frameworks and user acceptability. DRM typically affects information flows that are subject to copyright laws. What constitutes acceptable usage under copyright law depends subtly on context. Reliably determining and judging the relevant context remains a challenge for computers [10, 11]. Usage policies that DRM systems can enforce are usually much less nuanced and more restrictive than those of copyright law. Thus, many users resent DRM. On the contrary, EDRM typically affects information flows that are subject to NDAs and/or privacy laws and regulations, such as HIPAA or GLBA. Such agreements, laws, and regulations often stipulate strict usage policies and penalties for noncompliance. Both distributors and recipients of such information may welcome EDRM's ability to reduce risks of noncompliance [12].

Two previous projects have developed TPM support for Linux: Enforcer [13] and tcgLinux [14]. Neither project targeted usage controls, however. In Enforcer, a manifest can list what applications can run in the system and open specified files. However, the manifest needs to be signed by the system's trusted administrator. Thus, a file distributor cannot specify new usage policies that are enforced by a client unless the file distributor is also the client's administrator or convinces the latter to sanction them. In tcgLinux, the kernel extends into PCRs integrity measurements of all programs executed, in the order of execution. Thus, PCR values vary with execution order. Such variable PCR values are unsuitable for TPM sealing. If a UCFS's secret key is sealed to particular PCR values, it may not be possible to unseal the key reliably.

Lack of interoperability may hamper DRM and EDRM adoption. Prominent efforts for enhancing DRM interoperability include the MPEG-21 REL [15] and OMA DRM [16] standard rights languages, and the NEMO [17] system for orchestrating distributed services with heterogeneous media and rights formats. Like NEMO, and unlike MPEG-21 and OMA, our work supports interoperability using unmodified applications. We also provide solutions for enforcing usage policies securely on commercially available computers and usable interfaces for EDRM, which had not been described and evaluated in previous literature.

6. CONCLUSIONS

Usage controls enable the distributor of a file to limit how recipients of that file may use it. Usage controls can be negotiated and enforced automatically, quickly and inexpensively. Hence, they could be an atractive alternative or adjunct to non-disclosure agreements. However, existing usage controls often are software-based and easy to break. We designed and implemented operating system modifications that allow hardware-based usage controls to be easily added to existing software-based ones, hardening them. We also designed and implemented Web server and browser modifications for transmitting usage-controlled files securely. In this paper, we described our system's user interfaces and evaluated the system in a user study. In the study, untrained users role-played design engineers in two similar collaborative scenarios with or without usage controls. The results were very promising. Users found the interfaces easy to use and had insignificant trouble in considering documents' usage policies when deciding whether to accept them. They also had little trouble in authoring documents with appropriate usage policies. Usage controls had insignificant impact on the completion times and accuracy of the assigned tasks. These results suggest that the usage control approach described in this paper can add security to collaborative workflows with minimal training and performance penalties.

Acknowledgements

This project was funded in part by a grant from the NSF Center for e-Design [18].

7. REFERENCES

[1] OpenOffice, `http://www.openoffice.org/`

[2] Adobe. "Adobe Portable Document Format version 1.7", Nov. 2006. `http://www.adobe.com/devnet/pdf/`

[3] D. Touretzky. "Gallery of Adobe Remedies", `http://www.cs.cmu.edu/~dst/Adobe/Gallery/`

[4] D. Kyle and J. Brustoloni. "UCLinux: A Linux Security Module for Trusted-Computing-based Usage Control Enforcement", in *Proc. 2nd Workshop on Scalable Trusted Computing*, ACM, Nov. 2007, `http://www.cs.pitt.edu/~jcb/papers/stc2007.pdf`

[5] Trusted Computing Group. "Trusted Computing Platform Alliance (TCPA): Main Specification version 1.1b", Feb. 2002, `http://www.trustedcomputinggroup.org/`

[6] Foolabs. "Xpdf: A PDF Viewer for X", `http://www.foolabs.com/xpdf/home.html`

[7] C. Wright, C. Cowan, S. Smalley, J. Morris and G. Kroah-Hartman. "Linux Security Modules: General Security Support for the Linux Kernel", in *Proc. 11th USENIX Security Symp.*, Sept. 2002, `http://www.usenix.org/publications/library/proceedings/sec02/full_papers/wright/wright.pdf`

[8] R. Ianella. "Open Digital Rights Language version 1.1", Sept. 2002, `http://www.w3.org/TR/odrl/`

[9] J. Cohen. "Statistical Power Analysis for the Behavioral Sciences", Lawrence Erlbaum, Hillsdale, NJ, 1988.

[10] L. J. Camp, "First Principles of Copyright for DRM Design," in *IEEE Internet Computing*, May-June 2003, pp. 59-65.

[11] J. Erickson and D. Mulligan. "The Technical and Legal Dangers of Code-Based Fair Use Enforcement," in *Proceedings of the IEEE*, 92(6):985-996, June 2004.

[12] M. Donner. "Whose Data Are These, Anyway?," in *IEEE Security & Privacy*, May-June 2004, pp. 4-5.

[13] J. Marchesini, S. Smith, O. Wild, A. Barsamian and J. Stabiner. "Open-Source Applications of TCPA Hardware", in *Proc. Annual Computer Security Applications Conf.* (ACSAC), 2004, `http://www.acsac.org/2004/papers/81.pdf`

[14] R. Sailer, X. Zhang, T. Jaeger and L. van Doorn. "Design and Implementation of a TCG-based Integrity Measurement Architecture", in *Proc. 13th USENIX Security Symp.*, `http://www.usenix.org/publications/library/proceedings/sec04/tech/full_papers/sailer/sailer.pdf`

[15] X. Wang. "MPEG-21 Rights Expression Language: Enabling Interoperable Digital Rights Language," in *IEEE MultiMedia*, Oct.-Dec. 2004, pp. 84-87.

[16] W. Buhse and J. van der Meer. "The Open Mobile Alliance Digital Rights Management," in *IEEE Signal Processing*, Jan. 2007, pp. 140-143.

[17] R. Koenen, J. Lacy, M. Mackay and S. Mitchell. "The Long March to Interoperable Digital Rights Management," in *Proceedings of the IEEE*, 92(6):883-897, June 2004.

[18] NSF Center for e-Design, `http://www.e-designcenter.info/default.aspx`