LendLocked: Privacy & Transparency for Digital Library Lending

Boya Wang*
MPI-SP & EPFL
boya.wang@mpi-sp.org

Peter Hall*
New York University
pf2184@nyu.edu

Sunoo Park
New York University
sunoo.park@nyu.edu

Abstract

Digital library lending is a critical resource for access to information. Currently prevalent models of digital lending, however, involve opaque *licensing* schemes that entail serious drawbacks to reader privacy and freedom of expression. In popular modern library apps, publishers and hidden intermediaries control a wealth of information about readers and reading habits, at a scale and level of detail that would be essentially impossible in physical library lending.

To understand digital lending needs in practice, our work begins with a series of *interviews with library professionals* ($N = 11$). We present thematic findings on their concerns with existing systems, including privacy, surveillance, preservation, and lack of library control over resources. Many of the concerns raised are inherently unproblematic in the context of physical library lending—leading us to our central technical question: *Can digital lending achieve privacy and transparency at least as strong as physical library lending?*

Based on our qualitative findings, we provide the *first rigorous modeling* of security, privacy, and transparency requirements in digital library lending. As existing systems fall short of the strong guarantees we model, we propose a *new system design*, LendLocked, based on cryptography and trusted hardware, and prove it achieves these guarantees in the random oracle model. We *micro-benchmark our design's key cryptographic functionalities*, showing tolerable efficiency at the scale of the largest libraries.

Keywords

Privacy-Enhancing Technologies, Cryptography, Transparency

1 Introduction

Over history, various entities have attempted to intrude on reader privacy and access to information, sparking backlash, at the same time, reinforcing libraries' role as defenders of intellectual freedom and privacy against risks that are far from hypothetical [16, 48, 50, 56]. As an example, the Patriot Act expanded government access to library records after 9/11, and was described as a "danger to the constitutional rights and privacy rights of library users" [10].

A vital public service that libraries provide is lending. Libraries purchase and store, traditionally, physically copies of books. Library patrons can borrow limited numbers of these books for limited time periods, typically free of charge. Over recent decades, digital lending has emerged as a crucial means of expanding access to information—particularly in the COVID-19 pandemic [4]. In the last five years, digital lending has increased 34 percent [6].

This trend has also quietly introduced serious drawbacks to privacy, transparency, and library autonomy. In contrast to libraries' traditional values where patron privacy and transparent governance have long been foundational [26], current digital lending infrastructure is commonly built around opaque licensing agreements and commercial platforms whose incentives are to prioritize monetization via data collection over libraries' and patrons' interests. The data practices of these platforms are opaque not only to patrons but also often to libraries themselves [32].

Although these digital lending systems have drawn concerns from librarians, advocates, and scholars [22, 28], they remain widespread [51]. Prior scholarship highlights growing concerns over surveillance and data practices in digital lending: "although privacy is one of the core tenets of librarianship, technology changes have made it increasingly difficult . . . to ensure the privacy of their patrons in the twenty-first century library" [13]. These privacy risks are embedded in the infrastructure of modern digital lending.

In comparison, traditional library lending of physical materials offers strong privacy, transparency, and library autonomy which are remarkably hard to match in the digital context. When patrons borrow physical books, there is minimal tracking involved. The process does not require the collection of personal data beyond basic identification. Once a physical copy is checked out, it creates no (digital) footprint that could be used to track reading habits. Data collected is managed by libraries themselves, rather than through a complex network of publishers and third parties. Thus, physical lending is resilient to many concerns that emerge in digital systems where publishers and platforms have much greater capabilities of tracking users' interactions with content. Moreover, the tangible nature of physical lending provides intuitive transparency around data flows. Therefore, the central question in this work is:

Can digital lending achieve privacy, transparency, and access control at least as strong as physical library lending?

Though simple to state, even the question itself requires careful examination and contextual grounding towards a clear definition. Physical library lending has long relied on norms, institutional trust, and inherent qualities of physical (print) objects, rather than technical mechanisms or formal guarantees. Translating those practices into the digital realm raises novel challenges on definitions, threat models, and best-possible assurances to stakeholders, as "borrowing" now involves digital data and interactions mediated by software, vendors, and complex agreements.

To tackle this central question, we take an "end-to-end" approach[1] from formulation to modeling to implementation. Our

*Joint first authors.

Proceedings on Privacy Enhancing Technologies 2026(4), 83–113

https://doi.org/10.56553/popets-2026-0111

[1] Our approach follows the spirit of "cryptography for the people" [34] and human-centered design principles recognizing that systems often fail when we design without understanding the humans who will interact with them [33, 52].

work hence consists of three parts: (1) interviews & qualitative analysis, (2) model & definitions, (3) construction & evaluation.

First, to establish a baseline understanding of digital lending needs and concerns on the ground, we conduct a series of *interviews with library professionals* ($N = 11$). We perform thematic analysis and highlight notable concerns with existing systems; our analysis results in five central themes that inform our central technical question, model, and design.

Second, we provide the *first rigorous model and definitions of privacy, transparency, and access control requirements specific to digital library lending systems*, informed by the above qualitative findings. We translate core guarantees provided by physical library lending into precise technical requirements for digital lending systems and discuss how our model by design mitigates raised concerns.

Third, we propose a *new digital lending system design, LendLocked*, using cryptographic authentication and zero-knowledge proofs, alongside trusted hardware for deterrence of copying borrowed materials. We prove that LendLocked satisfies the strong privacy and transparency definitions in the random oracle model and micro-benchmark its key cryptographic functionalities. Our novelty lies in tailoring a combination of lightweight existing cryptographic objects for real-world privacy and transparency needs in this underexplored application setting, and proving security thereof. Mitigating stakeholders' concerns using lightweight tools with little overhead is in fact a requirement in this context, where library workers and patrons often have limited resources, as documented by our interviews in § 3. Our design relies on a library catalog supported by a cryptographic transparency log. Based on a rigorous threat modeling in § 5.3, this design can offer meaningful enhanced accountability for libraries' conduct. While the privacy shortcomings of current digital lending practices might seem unavoidable in the digital medium, our design demonstrates the feasibility of achieving security requirements without compromising privacy.

Summary of Contributions.

(1) We conduct *interviews with library professionals* ($N = 11$) and present detailed thematic findings (§ 3).
(2) We give the *first rigorous model* of private, transparent digital library lending (§ 4 and § 5).
(3) We propose a new *cryptographic system design*, LendLocked, and prove it achieves our strong definitions (§ 6 and § 7).
(4) Our *micro-benchmarks* demonstrate that our design is efficient at the scale of the largest libraries (§ 8).

We hope that our qualitative, theoretical, and practical cryptographic contributions can build a holistic foundation for future work on privacy and transparency in library lending—a comparatively under-studied area in security and privacy literature, which presents pressing and timely concerns [9, 25, 44].

2 Background & Related Work

Though the lending of physical books is simple to explain, digital library lending is not as straightforward a concept. What does it mean for libraries to "purchase" and "own" a digital copy of a book? What does it mean for patrons to "borrow" that copy and later "return" it? Implementations of digital lending can take diverse forms depending on one's answers to these questions.

Perhaps the most straightforward aspect to describe is the intended user experience. Much like with physical lending, patrons should be able to browse the collection of books their library offers for loan. When a patron "borrows" a book, they should gain access to its content for a set period, in a manner that supports a comfortable consumption experience, e.g., reading or reference. Once the book is "returned", the patron's access to that content should end. Despite the similarities in user experience, the library's perspective differs significantly between physical and digital lending.

Physical Collections. In the physical context, a library typically purchases physical copies of each book in its collection, which it then owns outright. This ownership grants the library broad control over how the purchased materials are circulated, including lending them to patrons without needing ongoing approval or oversight from publishers. The physical nature of the materials naturally limits simultaneous access and prevents concurrent loans, limits surveillance of patrons during the loan period, and supports straightforward returns by physical transfer.

Licensing. Digital copies of books generally cannot be purchased and owned in the same way as physical copies. Instead, libraries generally *license* access to digital content under terms often set by publishers and intermediaries called distributors [51]. A license is a contract that grants a library limited rights to access and distribute digital content under specific conditions, in exchange for payment from the library to publishers, distributors, and/or aggregators. Common types of licenses include *metered access*, where licenses expire after a number of checkouts or period of time, and *pay-per-circ*, where libraries pay each time a book is borrowed [51].

At least in these respects, digital lending shifts control away from libraries and introduces complex questions about ownership, access, as well as long-term stewardship [29]. Some of these issues are not inherent to licensing, but are rather features of the prevalent incentives, industry practices, and structural power imbalances in today's library lending landscape. For instance, while licensing does not necessarily require reliance on intermediary distributors or the use of proprietary e-reading platforms (see § C), such arrangements are commonplace in today's digital lending systems.

Distributors. Distributors and aggregators act as intermediaries between publishers and libraries, managing licensing agreements and access platforms, and often controlling how content is delivered, used, and tracked in digital library lending systems [51]. One major distributor is OverDrive, which has around 90% market share [51].

Controlled Digital Lending. A notable alternative method of digital lending is called *controlled digital lending* (CDL) [30], in which libraries digitize physical books they own and lend out the digital copies while maintaining a 1:1 "owned-to-loaned" ratio, i.e., ensuring that at most one copy of any given physical book owned by the library is loaned out at any given time, thus simulating the natural constraints of physical loans, and trading ongoing fees for an upfront purchase cost. Though it is well established law that the owner of a physical copy of a book is entitled to loan that physical copy out, publishers have contested the practice of CDL in a prominent recent lawsuit,[2] arguing that ownership of a physical

[2] *Hachette v. Internet Archive*, 115 F.4th 163 (2d Cir. 2024).

book copy does *not* entitle the owner to scan it and loan it out digitally, and that CDL is problematic for copyright holders because it enables widespread distribution different in kind from digital lending including distribution beyond the 1:1 ratio. We note that the transparency properties we define and realize in our design enable cryptographic verification that a 1:1 ratio is maintained in CDL and similar systems, without compromising patron privacy.

Other Related Work. Previous work in library and information sciences extensively documents the history and importance of patron privacy in libraries, alongside increasing concern about surveillance and data practices as digital lending (based on licensing) gains adoption [36]. These studies point to the ways in which current systems can undermine core library values not limited to privacy, such as freedom of inquiry, and equitable access. Some studies moreover connect privacy and transparency in libraries [31, 43].

Prior work in computer science, and specifically security and privacy, has been limited. A notable recent work examines how librarians perceive and navigate challenges related to patron privacy [45]. An earlier paper, from before digital lending, studied security/privacy in library RFID [47]. Another recent computer-security work studies privacy in library-provided IT services, which is largely distinct from loan privacy [42]. Another strand of computer-science literature focuses on digital rights management (DRM) techniques, not limited to libraries (e.g., [1]). While DRM is arguably orthogonal to privacy and transparency, DRM implementations often introduce access control and tracking mechanisms. They also often obscure their inner workings [54], negatively impacting privacy and transparency. When DRM systems are insecure, yet further privacy issues may arise [7].

3 Interviews

Our interviews serve to develop a working understanding of library professionals' goals and constraints around digital lending. The interview study was approved by our institutional review board.

3.1 Methodology

From December 2024 to December 2025, we interviewed 11 U.S.-based library professionals with expertise across digital licensing, digital lending, digital preservation, collections, library administration, and patron services. Table 4 shows the background information of all interviewees with randomly assigned participant numbers. All interviewees have 10+ years experience in fields related to libraries or digital lending. To respect participants' choices on anonymity and consider the potential adverse privacy consequences, we do not release participant identities nor transcripts.

Recruitment & Participants. Participants were recruited by outreach to mailing lists within relevant communities, by outreach to the authors' professional networks, and by snowball sampling. We broadly solicited participants with expertise in library lending and/or library technology without monetary compensation.

Interview Protocol. We conducted semi-structured interviews based on the protocol in § D.8. Interviews took place via Zoom video call and mostly lasted about one hour. One author led the questioning across all interviews with at least another author present. Permission to record audio was not required for participation. However, all interviewees granted this permission. Therefore, all interviews were audio recorded in this study.

We began by asking participants about their experience and expertise. We grouped subsequent questions into topic areas such as (digital) lending practices & workflow, patron information & loan records, library policy violations, and publisher interactions. In each interview, we chose which topic areas to focus on, depending on the participant's stated experience and expertise.

We stated upfront that our research focused on privacy and transparency in digital lending. While our protocol include detailed questions about data practices, it also intends to draw out experiences and concerns broadly related to digital lending technologies, not limited to privacy and transparency, to enable participants to raise unanticipated topics not framed as "data/design considerations." This flexible approach proved fruitful, as we learned about a range of practitioner concerns beyond our anticipated scope.

Data Analysis. We created initial transcriptions of audio recordings using Zoom's transcription feature. One author reviewed the recordings and transcriptions to check for accuracy, then conducted reflexive thematic analysis [14]. Another author reviewed the results of analysis in detail.

Scope Limitations. Our focus was on librarians' perspectives; we did not interview patrons, nor publishers or distributors. Librarians are concerned about patron interests, but lack patrons' firsthand perspective. We limit our study to librarians both to keep scope manageable, and as we believe librarians are more likely to have expertise in library technologies. Publishers' and distributors' interests may be in tension with libraries' interests, as indicated in prior library and information sciences literature (see § 2). Studies on other stakeholders' interests would be fruitful future work.

3.2 Thematic Findings

We now present our thematic findings on existing approaches to digital lending and the need for new approaches. Due to limited space, § D offers additional detail on most themes.

3.2.1 Thematic Findings That Inspire Our System Design. We discuss 5 themes related to existing approaches to digital lending, and then, briefly, 3 themes about the need for new approaches.

Theme 1: Complementarity of physical and digital lending. Almost every participant involved in procurement (P3, P4, P7, P8) affirmed that digital lending is very popular with patrons, and that digital formats can better suit some patrons' needs, often for reasons of accessibility (e.g., large print needs). At the same time, we heard that physical formats are a better fit for many patrons, often also for accessibility reasons. For example, "*I'm old, I'm visually impaired. I want to hold a book in my hand.*" (P8). P4 and others noted that this can be personal: "*[S]ome people prefer print. I know I prefer print.*" In sum, digital and physical lending each serves different needs, and digital lending is an essential offering for modern libraries.

Theme 2: Lack of library control. The limited control that libraries have over their digital offerings was a recurring theme: control over content, over data flows, and over technology choices.

> "*I think the big limitations for most libraries are, like I said, they don't host their own content. [T]hat has less to*

> *do with resources and more to do with knitting together the necessary technology. … There are real strong incentives for libraries to figure out some way to actually host and own their own copies …*" —P1

And yet such technology offerings are very limited. P6 highlights cost and scalability as countervailing considerations against self-hosting, but agrees on the concerns around third-party platforms:

> "*Whoever controls the reader controls the data flow. … There's always a possibility that the data could flow …for their benefit, and not for our benefit.*" —P6

Theme 3: Opacity of technologies and licensing. Multiple participants expressed discontent over the opacity of library technologies and of licensing schemes. For example, P2 expressed frustration about the "*black box*" nature of "*digital intermediaries*" such as Overdrive, and that "*figur[ing] out how much data is going in and out*" of such platforms is "*nearly impossible.*" P10 described pricing as "*opaque*" and voiced frustration at platforms rolling out new features without notice or choice for libraries: "*We didn't know that [Kanopy was highlighting a collection of content that some find morally questionable] ahead of time. We didn't have the chance to opt in or out.*" P6 noted that their institution is "*trying to move … to … open standards, interoperable standards.*" Multiple participants (P1, P2, P4, P5, P9) noted that users are often unaware of how their data is kept and processed, and may not be in a position to opt out.

Theme 4: Data integrity and preservation. Five participants (P1, P6, P8, P10, P11) raised specific concerns about data integrity and preservation. Three participants (P2, P5, P7) raised concerns about censorship and book bans. Some of these concerns relate to libraries' lack of control over content (Theme 2). P11 discussed preservation at the most length, expressing concerns around "*what happens when [a] vendor goes bankrupt or … when they decide that they don't care about this part of their business anymore,*" noting that they "*[ha]ve seen this actually a couple of times*" and explaining that "*if libraries are supposed to fill this role of maintaining a … cultural record that has some integrity to it, then they really [need to] have an adequate level of control over copies of that record.*" P11 also discussed their institution's home-grown system and routine fixity checks that confirm the absence of corruption or attack—on a roughly monthly basis through a constant background process, due to the size of their operation (a fact we reference in our evaluation).

Theme 5: Surveillance and privacy. All but three participants expressed specific concerns about privacy or surveillance in digital library services. P5 expressed concerns about the longevity of lending records, noting that "*that's something that I don't really like about our system,*" as well as concerns about insider abuse of these records, based on a real incident they had heard about. P9 notes that there is "*still a fair amount of diversity*" in library data policies, "*despite the … widespread understanding of the threats to user privacy around the collection and retention of lending data.*" And P2 raised concerns about companies' incentives when faced with the promise of a "*panopticon of reader usage*" in a context where it may seem that "*if you have more data, then you will make more money.*" They added that despite the seeming promise, "*in reality that's generally false.*" In contrast, P4 was personally unconcerned about data collection by distributors, "*Personally, no, I don't think it's an issue*", but saw data protection as important for compliance with library policies.

On the flip side, four participants (P1, P5,P6,P9) discussed that data about reader preferences and history has utility both for libraries and for users, which can be in tension with the privacy drawbacks of retaining it. From the patron perspective, P1 observed, "*it's sometimes pretty nice to log into a system and … remember the stuff that you read before*", and that "*some users love that*", "*but [that] also means now that vendor knows all about what you are doing …*."

And from the libraries' perspective:

> "*[L]ibraries like to know how their books circulate, and they like to know a lot about the collections themselves. But for the most part they're able to do that without tracking a whole lot. I mean, knowing that a book circulated is a lot different than knowing who borrowed it, and how long and what they did with it. And so the basic approach across libraries for a very long time has been: we'll track when stuff comes in and when stuff goes out, but we're not going to keep track of anything more than what we need. … [L]ibraries have really buckled down on just saying we would prefer to not know or not retain records [due to privacy considerations].*" —P1

P1 described the above approach as heavily influenced by "*a pretty long history of*" libraries dealing with records requests from "*law enforcement or by other groups*" interested in "*finding out … What kind of books did they look at? And, you know, other things like that.*" All four participants here mentioned that collecting additional information would likely be "*useful, … for making collections decisions, for instance*" (P1), for "*educational and scholarly*" purposes (P6), and for promoting underutilized resources to certain patrons.

> "*There are, as in so many other cases, potential benefits in knowing what user behavior is like and providing enhanced services. …So you know, if I know what user populations, at least at some level of granularity, were consuming what, then I could better tailor my services to meet [their] needs …*." —P9

At the same time, all four participants expressed serious privacy concerns about retaining much of that kind of data. While P5 notes that libraries they are familiar with *do* sometimes retain this kind of data, P1 "*think[s] libraries have been very hesitant about it, because of their [privacy] concerns.*"

> "*[T]he top priority is like, let's not screw up people's privacy along the way. And if we can learn interesting things that's great, particularly around what we should acquire in the future.*" —P1

Need for new technologies and approaches. Participants expressed a keen interest in new technological approaches that improve privacy (Theme 6) and accessibility (Theme 7), and shift the market landscape away from existing digital licensing models (Theme 8). P2 emphasized the importance of "*a culture of ownership for [library] institutions and choice in the market*", and P9 remarked:

> "*If there were … privacy protecting mechanisms that we are presently unaware of we would very much appreciate considering them …*" —P9

Additional discussion on all interview themes can be found in § D, including a range of rich detail that did not directly inform our design, but which highlights pressing practical problems and promising ground for future work such as privacy concerns around personalization of library services, and concerns with AI-generated content and AI features on library technology platforms.

4 Motivating Our Model

We design a new rigorous model of digital lending, guided by the thematic findings in § 3.2.1. Theme 1 demonstrates the need for, and benefits of, digital lending within modern library ecosystems. Yet Themes 2–5 highlight a range of pressing practical concerns with currently available digital lending solutions, from libraries' perspectives. Finally, Themes 6–8 emphasize the need for new solutions that offer privacy and transparency while exploring alternatives to existing licensing models. Therefore, we ask: Could a new approach to digital lending (Themes 6–8) achieve similar benefits (Theme 1) while mitigating or resolving these practical issues related to privacy, transparency, and library control (Themes 2–5)?

Then, observing a common property of Themes 2–5—none is an issue in physical lending—we broaden our question slightly:

Question 1. Can digital lending achieve privacy, transparency, and library control at least as strong as physical library lending?

A notable prior effort with similar stated goals is controlled digital lending (CDL) as mentioned in § 2. However, the CDL white paper was a high-level practical proposal. Unlike this work, it neither aimed to give rigorous models and definitions, nor a qualitative study. CDL was deployed by the Internet Archive, and became the subject of a legal controversy where publishers argued influentially that its access control measures were insufficient to ensure the 1:1 owned-to-loaned ratio that the system promised (see § 2). Learning from this, we add one more desideratum, i.e., our new approach to digital lending must ensure at least as strong DRM-style access control as current licensing systems.

For access control only, we treat "as similar as possible to digital licensing" as an acceptable fallback benchmark in situations where achieving properties similar to physical lending is impossible. This modeling choice is motivated by our aim of mitigating privacy concerns while demonstrating compatibility with publishers' stated concerns; clearly, the access control standards prevalent in digital licensing are a solution that is acceptable for publishers in practice. "As similar as possible to digital licensing" would *not* be an acceptable fallback standard for privacy as the point of our privacy design is to mitigate participants' concerns with digital licensing.

Question 2. Can our new digital lending design moreover achieve access control as strong as existing digital licensing systems?

Our answer is *yes* to both questions. In § 4–§ 5, we offer a new model and definitions for digital lending. In § 6–§ 8, we propose a system protocol and prototype implementation that provably meets our security definitions with acceptable efficiency.

Much of our modeling is guided by properties of physical lending. Ultimately, § 5.4 connects our model and definitions back to our initial desiderata: namely, those based on Themes 1–5 and ensuring access control comparable to existing digital licensing systems.

4.1 Physical Lending

A physical lending system achieves the following properties.

Property 1 (Privacy)**.** The publisher of a book cannot observe other parties lending and accessing that book after an initial purchase. The library that owns a book cannot observe usage and access patterns of that book while it is lent out.

Property 2 (Exclusivity)**.** Any given copy of a book cannot be simultaneously in the possession of multiple people. [3]

Property 3 (Returnability)**.** Upon returning, the patron loses access to the returned copy of the title.

Property 4 (Rate Limiting)**.** Among the simplest ways to access the content of a book is physical page by physical page.

Property 5 (Copying Deterrence)**.** Among the simplest ways to copy out a book are to write them down or to scan each page.

As a form of access control, Property 2 (Exclusivity) and Property 3 (Returnability) ensure that the library complies with the one-copy-one-loan mode in their lending service. In addition, Property 4 (Rate Limiting) and Property 5 (Copying Deterrence) introduce considerable frictions to the potential attempt of pirating the borrowed copy. As a result, the risk of large-scale piracy decreases to an acceptable level of copyright holders, given the benefits of information dissemination in library lending service.

4.2 Licensing

As Property 2 (Exclusivity)–Property 5 (Copying Deterrence) are inherent to the physical medium and difficult (or practically impossible) to emulate digitally, existing digital licensing schemes utilize other properties below, valued by publishers and rights-holders. These properties ensure that libraries cannot loan out the same copy of a book to multiple people at once, at the same time, achieve practically limiting access and copying.

Property 6 (Auditability)**.** A third party can check or enforce that libraries adhere to an allotted limit on digital loans.

Property 7 (Access Restrictions)**.** It is difficult to access content outside predefined queries.

Auditability substitutes for Property 2 (Exclusivity)–Property 3 (Returnability), as a third party can ensure that the library is not overlending. DRM is often used to limit access as a substitute for Property 4 (Rate Limiting)–Property 5 (Copying Deterrence).

4.3 Direct vs Intermediated Systems

We distinguish between *intermediated* and *direct* digital lending systems (see § E). In intermediated systems, another party has control over the content transferred and mediates the interactions between a library and a patron. In direct systems, libraries manage storage and access to the content they lend.

Our model and construction (§ 5–§ 6) are designed for *direct* digital lending. We choose to focus on direct lending because (1) it is more responsive to the expert concerns related to lack of library control over digital lending systems (Themes 2, 3, 4) and (2) precisely

[3] N.B. This definition permits re-loaning: if a person re-lends a book to another person, this person no longer has the book while it's in the other one's possession.

because it is rarer in practice, its exploration is important to shift the policy debate as discussed above. Some of our ideas could also apply to intermediated systems (see § C); a detailed extension of our work to intermediated settings (including licensing-based schemes) would be valuable future work.

4.4 Design Scope & Intention

Why Stop at Physical Lending? While physical lending is not perfect, it embodies the standards of privacy and transparency for library contexts, which have been societally acceptable for far longer than digital lending has been around. Moreover, physical lending has remarkably strong privacy and transparency properties. In fact, achieving commensurate guarantees in the digital context presents non-trivial challenges, in some cases even impossible. That said, conversely, digital techniques may enable even stronger guarantees than physical lending in certain respects, e.g., enabling constraining library access to data even beyond the physical setting. Exploring such stronger guarantees would be fruitful future work.

Shifting the Policy Debate. As shown in the following sections, our modeling of access control alongside the other desiderata demonstrates that enforcing access constraints need not involve publisher control of systems, nor sacrificing patron privacy. By providing concrete constructions and formal proofs based on the modeling, we show possibility of achieving *privacy, transparency, and access control commensurate to physical lending* in the digital setting in a meaningful, rigorous way—a novel concept that is non-trivial even to define, and the possibility of which has been subject to prominent doubts.[4]

In current practices, publishers can reap other benefits from the licensing model and control over systems in practice, beyond access control. Our proof of concept may not remove all obstacles to publishers embracing privacy-preserving, library-controlled digital lending. However, we hope to help shift the broader discourse away from the framing of publisher control of systems, and the licensing model, as a necessity for access control in digital library lending.

5 Our Model

We now present the assumptions, entities, syntax, threat model, and definitions that constitute our model.

Communication & Clock. We assume that entities communicate via authenticated channels, except for user-reader pairs which communicate via direct physical connection. Since libraries grant loans within time limits, we assume that all entities can query a clock oracle $C(\cdot)$, which provides a consistent view of time.

Entities. A digital library lending system involves these entities:

- **User ($\mathcal{U}$)**, a patron requesting and accessing loans using the library lending service.
- **E-Reader ($\mathcal{R}$)**, a reading device that can load and give patrons access to loaned contents.
- **Library ($\mathcal{L}$)**, an entity preserving and managing loans to provide lending service to patrons. To fulfill this role, $\mathcal{L}$ maintains a public append-only log logBB containing catalog data, e.g., number of available copies of a title. [5]
- **Auditor ($\mathcal{V}$)**, any entity who verifies that $\mathcal{L}$ provides the lending service in a controlled manner. E.g., a publisher may be motivated to check that $\mathcal{L}$ does not over-loan.

Trusted Hardware. We model the e-reader $\mathcal{R}$ as a trusted execution environment (TEE): $\mathcal{R}$ only performs routines and protocols as described, and the memory of $\mathcal{R}$ may be split into $\mathcal{R}_{pub}, \mathcal{R}_{priv}$ where the user can access $\mathcal{R}_{pub}$ but not $\mathcal{R}_{priv}$. We provide more details and justifications on this modeling choice in § 5.3.

Library Policies. We treat the library as trusted to enforce certain policies that are typically administered at libraries' discretion and may involve non-digital processes, e.g., setting and checking eligibility requirements for patrons or consequences for late returns. We abstract such policy enforcement as black-box procedures, and stress that these library policies are not enforced cryptographically.

Availability. $\mathcal{U}$, $\mathcal{R}$, and $\mathcal{V}$ may go offline any time. We assume $\mathcal{L}$ to be online at all times, and logBB is always available to all parties. We treat whether patrons are permitted to read digitally borrowed books while offline as a matter of library policy. If not, then automatic returns can be consistently enforced once deadlines hit by requiring $\mathcal{R}$ to check the current timestamp is in line with the deadline whenever $\mathcal{U}$ wants to access part of the borrowed content. In practice, existing digital lending applications do sometimes permit offline access to borrowed materials. In this case, patrons may be able to keep accessing books past the deadline by going offline, unless the e-reader enforces the deadline using a local clock. But the policy violation would be evident to the library as no return operation would occur by the deadline. As soon as the user came back online, which is necessary to continue using library services, the library could impose any consequences.

5.1 Workflow & Syntax

The workflow contains five high-level interactive protocols: Register, Borrow, Access, Return, Audit. Next, we specify the syntax for sub-protocols under each protocol. We indicate involved entities by appending them at the end of the syntax.

Register. Before taking out loans from $\mathcal{L}$, $\mathcal{U}$ must sign up with $\mathcal{L}$ as a patron. To access loaned contents digitally, $\mathcal{U}$ must also register a digital reading device $\mathcal{R}$ under their account with $\mathcal{L}$.

- patronregUL takes as input some identity information u.info of a new user, and returns a user registration key k_u.
- readerregURL takes as input k_u, a reader key k_r, and derives a new key for this user-reader pair k_{ur}.

Borrow. After registration, $\mathcal{U}$ can request loan of a title b.title from $\mathcal{L}$. Upon approval of this loan request, $\mathcal{L}$ appends a loan record on logBB. $\mathcal{L}$ also sends the content of b.title together with its return deadline information to $\mathcal{R}$ registered under $\mathcal{U}$'s account.

- borrowreqUL takes as input a borrow request of a title b.title. $\mathcal{L}$ verifies the legitimacy of this request from $\mathcal{U}$ and the availability of b.title. Upon successful verifications, $\mathcal{L}$ approves this request and updates the loan status on logBB.

[4] As mentioned before, prominent arguments in the *Hachette* case suggest the status quo of surveillance and control in digital lending may be necessary for access control.

[5] Because logBB is public, we assume all protocols may take logBB as additional input. For readability, we only explicitly state this for auditing protocols.

- addlogURL takes as input the successful borrow request include a timestamp of the request ts, patron ID uid, book name b.title, k_{ur}, and a signature σ from $\mathcal{R}$. $\mathcal{R}$ signs the request, and $\mathcal{L}$ appends the request and the signature on logBB as a borrow record; we treat the appended record as the output of addlogURL.
- downloadRL takes as input a requested title b.title. $\mathcal{L}$ verifies that $\mathcal{U}$ associated with $\mathcal{R}$ has successfully borrowed b.title, then sends the content of b.title. $\mathcal{R}$ runs a subprotocol verilogR described below.
- verilogR is run by $\mathcal{R}$ to check consistency of the newly added record on logBB and $\mathcal{R}$'s previous borrow request. Upon successful verification, $\mathcal{R}$ caches the borrowed content according to that request locally.

Though running as a subprocedure of the Borrow protocol in our construction, we later refer verilogR as *self-auditing* from the perspective of its functionality. verilogR enables each $\mathcal{R}$ to check that the private information in their local state about its own borrow/return requests are correctly reflected on logBB. In principle, $\mathcal{R}$ can run verilogR as often as they want. However, given the log is append-only, it is sufficient to check each time the log should be updated to reflect an action they take.

Access. After borrowing the intended title b.title and downloading its content, $\mathcal{U}$ can access the content using $\mathcal{R}$ within the loan period. During access of borrowed content, $\mathcal{R}$ also updates its internal state.

- readUR takes as input k_u, b.title, and an index i of b.title. $\mathcal{R}$ looks up the referred part of the content b.cont$[i]$ of b.title which has been stored locally. Otherwise, $\mathcal{R}$ aborts.
- trackreadR keeps track of the access progress (e.g., reading page, text copy speed etc.) by updating $\mathcal{R}$'s internal state.

Return. $\mathcal{U}$ can initiate the return of a loan on $\mathcal{R}$ back to $\mathcal{L}$ before the return deadline. When returning, $\mathcal{R}$ deletes any locally cached content of b.title. Upon receiving a return request, $\mathcal{L}$ appends a corresponding return record on logBB. Without user initiation, $\mathcal{R}$ can also run the return protocol automatically with $\mathcal{L}$ at the first time when $\mathcal{R}$ goes back online passing the return deadline.

- returndelURL takes as input a borrowed and cached title b.title. $\mathcal{R}$ produces a signed return request to $\mathcal{L}$. Upon successful verification of the return request, $\mathcal{L}$ marks this title as returned, and calls addlogURL to append a return record on logBB. $\mathcal{R}$ runs a subprotocol verilogR to ensure consistency of the newly added record on logBB and the previous return request. Upon successful verification, $\mathcal{R}$ deletes the locally cached content b.cont of b.title.

Audit. Any entity can act as $\mathcal{V}$ to check the loan activities of $\mathcal{L}$ at any time using the public information on logBB. In addition to logBB, all parties may view DistLst, which serves as a set of "promised" copies of each book b.title the library has procured from the distributor(s).

- auditlogV takes as input all the records on logBB. $\mathcal{V}$ verifies the validity of each record, e.g., by verifying cryptographic proofs and signatures. Upon successful verifications, $\mathcal{V}$ cross-checks the records with the number of copies of each title that $\mathcal{L}$ may loan out. If the loan activities pass these checks, the audit passes; otherwise, it fails.

Table 1: Syntax Mappings Across Three Tiers. First two columns show protocols and sub-protocols defined in our model (§ 5.1). Last column shows the corresponding procedure from our construction and algorithm pseudocode (§ 6).

Model & Definitions		**Construction**
Protocol	Sub-protocols	Procedures
Register	patronregUL	PatronRegL()
	readerregURL	GenRegReqU()
		KeyGenRegR()
		ProcessRegReqL()
		FinishRegR()
Borrow	borrowreqUL	PlaceLoanU()
	addlogURL	GenLoanMsgU()
		GenLoanReqR()
		ProcessLoanReqL()
	downloadRL	VerLoanResR()
Access	readUR	PrepAccessU()
		GenLoadReqR()
		ProcessLoadResR()
	trackreadR	ProcessLoadResR()
Return	returndelURL	PrepRetU()
		GenRetReqR()
		ProcessRetReqL()
Audit	auditlogV	EvalAuditProofV()
	verilogR	SelfAuditR ()
		VerLoanResR()
		ProcessRetResR()

Tiers of Syntax. Table 1 summarizes the protocols above, alongside some additional syntax (last column) that will be introduced in our construction in later sections.

5.2 Definitions

Using the syntax established in § 5, we specify requirements for digital lending systems. We organize our security properties into three groups by functionality: (1) registering, borrowing, and returning, (2) accessing/reading, and (3) auditing. We additionally include a definition for "copy deterrence", detailed below. In all definitions, adversarial parties are assumed to be probabilistic poly-time (PPT), to behave arbitrarily and possibly collude with one another, and to have access to the clock oracle and bulletin board. Table 2 overviews which parties are treated as honest or malicious for each definition.

Definition 1. Let LendLocked =

(patronregUL, readerregURL, borrowreqUL, addlogURL, downloadRL, readUR, trackreadR, returndelURL, auditlogV, verilogR)

be a tuple of efficient algorithms, let $\mathcal{L}$ be a party that implements their respective algorithms according to their syntax, and let $\mathcal{R}$ and $C(\cdot)$ be machines which each implement their respective queries. Then, we say that LendLocked is a *private, transparent digital lending system* if the appropriate algorithms of LendLocked realize

the register-borrow-return functionalities (Def. 3), the read functionality (Def. 4), and the audit functionality (Def. 5), and (readUR, trackreadR) satisfies copy deterrence (Def. 19).

A private and transparent digital lending system must satisfy all the following definitions. Due to space, we present them informally here[6] and refer to § G for full formal definitions.

Informal Definition 6. *Borrowing and reading correctness* requires that if (1) $\mathcal{U}$ is registered with $\mathcal{L}$, (2) $\mathcal{U}$ has borrowed a copy of b.title, and (3) the relevant loan period is not expired, then $\mathcal{U}$ can read b.title using the readUR protocol.

Informal Definition 7. *Borrowing soundness* requires that malicious users and readers not registered with an honest $\mathcal{L}$ cannot borrow books.

Informal Definition 9. *Returning correctness* requires that a returned book results in the book being returned according to logBB.

Informal Definition 10. *Returning soundness* requires that malicious users cannot continue reading a borrowed book after it has been returned as according to logBB, assuming the e-reader and the library initiated the return honestly (and can be malicious afterwards).

Informal Definition 11. *Borrowing and returning privacy* requires that logBB entries about a user's loans does not leak any information about that user to an outside adversarial auditor.

Informal Definition 12. *Reading soundness* requires that malicious users and libraries cannot load the content of books they do not have borrowed, so long as the e-reader is honest.

Informal Definition 13. *Reading privacy* requires that no outside malicious auditor (including, e.g., the library) can tell when and whether $\mathcal{U}$ performs reading actions on books.[7]

We support two types of audits: auditlogV checks logBB for internal consistency based on public information only, and verilogR, run by e-readers, checks logBB for consistency with private records.

Informal Definition 14. *Auditing correctness* requires auditlogV and verilogR to successfully verify honestly generated log entries.

Informal Definition 15. *Auditing soundness* requires that (a) no entry on logBB could be verified as a valid loan record for two distinct users, (b) logBB is append-only, and (c) that no reader will parse an entry as meant for them when it is not. In aggregate, this ensures that the public audit log must list one distinct entry for each distinct loan, and each property here is guaranteed separately—see § G for more details. In these games, all parties may be malicious other than the auditor.

Whistleblowing. In LendLocked, e-readers regularly perform checks to make sure that all material that their user has access to at a given time are represented by open loans (i.e., a borrow entry not followed by a corresponding return entry) for their user on logBB. Auditing soundness ensures that in these checks, no two readers can parse the same entry as representing loans for distinct users. As an optional feature, e-readers may be programmed to "whistleblow" if a mismatch is detected, indicating that the library is overloaning (i.e., providing access to materials without logging them in logBB). Whistleblowing would consist of alerting their user and/or reporting to auditors that a mismatch has been detected at library $\mathcal{L}$, without revealing any user information. See "Library catalog falsification" in §5.3 for more discussion.

Informal Definition 18. *Auditing privacy* requires that no adversarial auditor $\mathcal{A}$ can learn anything more from the publicly accessible logBB than the titles and timestamps of books borrowed or returned. (In particular, $\mathcal{A}$ cannot tell who borrowed what.)

Note that our auditing privacy notion matches the physical setting, in that a motivated adversary who checks the physical library's public catalog repeatedly would be able to learn this same information (but not who is borrowing or returning books). We give one example formal definition, for auditing privacy.

Definition 18. Let $k, \lambda \in \mathbb{N}$, and let $\{(\mathsf{ts}_i, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)\}_{i\in[k]}$ be a sequence of k credentials, and let logBB be the bulletin formed by appending $\ell = \mathsf{addlogURL}(\mathsf{ts}_i, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)$ in sequence. Let $\mathsf{info} = \{\mathsf{b.title}_i, t_i\}_{i\in[k]}$ be the titles borrowed/returned and timestamps of each record. We say that our scheme satisfies *auditing privacy* if, for all PPT adversaries $\mathcal{A}$, there exists a simulator Sim such that $REAL(\lambda) \approx SIM(\lambda)$, where

$$REAL(\lambda) = \{out \mid out \leftarrow \mathcal{A}(\mathsf{logBB})\} \text{ and}$$
$$SIM(\lambda) = \{out \mid out \leftarrow \mathcal{A}(\mathsf{logBB}), \mathsf{logBB} \leftarrow \mathsf{Sim}(\mathsf{info})\}.$$

Copy Deterrence. Finally, we briefly discuss copy deterrence, an auxiliary definition designed to model the practical difficulty of copying books in the physical setting — a moderate difficulty notion, as copying *is* always possible, but onerous. Our notion of copy deterrence takes inspiration from Property 5 (Copying Deterrence) and Property 7 (Access Restrictions) from § 4. Copy deterrence is not central to our work, whose main focus is privacy rather than DRM-style access restrictions (see § 4), but we found modeling copy deterrence to be quite interesting and nuanced; see § G.4.

5.3 Threat Model

Table 2 describes our threat model. As standard, correctness definitions assume all relevant parties are honest. All adversarial parties are by default assumed to be PPT and may be allowed to collude.

Privacy. Our privacy properties aim to ensure confidentiality of records that in traditional physical lending are known only to patrons and libraries, against third parties including auditors. Hence, users, e-readers, and libraries are trusted and auditors are untrusted. While we aim to emulate physical library lending privacy properties as closely as possible, some aspects are not exactly the same. Notably, in our definitions and LendLocked, the auditor can learn the exact timestamps and titles of each book borrowed and returned until the present moment. While monitoring of the library catalog can achieve a similar effect for physical lending, the amount of history that can be learnt depends on library policy about disclosing historical loans. Alternatively, *continuous* monitoring of a library catalog could achieve a similar effect, but especially before the advent of online catalogs, this would have been very onerous

[6] Informal definition numbers match formal definition numbers, and can be clicked to jump to the formal definition in the appendix.

[7] Reading privacy is trivial in non-intermediated systems, including physical lending. But intermediated systems, such as common licensing-based systems, fail to satisfy it.

Table 2: Threat Model Overview. ✓ means treated as honest; ✘ means treated as adversarial; – means the entity's behavior is not relevant to the definition (i.e., technically, it may be adversarial).

Functionality	Property	Def.	$\mathcal{U}$	$\mathcal{R}$	$\mathcal{L}$	$\mathcal{V}$
Borrow & Read	Correctness	6	✓	✓	✓	–
Borrow	Soundness	7	✘	✘	✓	–
Return	Correctness	9	✓	✓	✓	–
Return	Soundness	10	✘	✘	✘	–
Borrow & Return	Privacy	11	✓	✓	✓	✘
Read	Soundness	12	✘	✓	✘	–
Read	Privacy	13	✓	✓	–	✘
Audit	Correctness	14	✓	✓	✓	✓
Audit	Soundness	15	✘	✘	✘	✓
Audit	Privacy	18	–	✓	✓	✘
Read	Copy Deterrence	19	✘	✓	–	–

for physical lending. We believe the leakage inherent in our notion of digital lending is roughly comparable to that from continuous monitoring of online library catalogs.[8] While we do not see the history of book titles and timestamps of past loans, without any patron information, as particularly sensitive, we note this point for completeness, to thoroughly represent the differences between information revealed in our model versus in physical lending.

Soundness. Users and e-readers of non-patrons may collude to try to borrow a book to violate ***borrowing soundness***. Similarly, users may try to access content *that they have not borrowed* or *that they have returned*, to violate ***reading soundness*** or ***returning soundness*** respectively. So, $\mathcal{U}$ is untrusted; $\mathcal{R}$ is trusted for just reading soundness (as discussed under "Trusted hardware" below).

The library is trusted for borrowing soundness, but not for reading or returning soundness. For borrowing, this reflects libraries' discretion over who registers as a patron. For reading and returning soundness, overloaning beyond what is recorded in the library catalog is not possible thanks to the two italic conditions above.

The maintenance of catalog records is ensured by ***auditing soundness***, where the library and users are treated as adversarial. E-readers and auditors, who are trusted in this context, work together to check (respectively) that (1) borrow/return records on the public bulletin board are consistent with e-readers' private records, and (2) all records on the public bulletin board are consistent with the library loaning only a certain number of titles with only one loan per title at any given time.[9] Additional discussion on catalog maintenance is given under "Library catalog maintenance" below.

Trusted Hardware Discussion. Trusting the reader where the user is also trusted is standard,[10] so let us note the only two cases of divergence: ***reading soundness*** (Def. 12) and ***copy deterrence*** (Def. 19). Informally, ***reading soundness*** requires users cannot access books on which they do not have an open loan, and is necessary for *returning soundness*, which requires that users cannot access books they have returned. In current digital lending systems, this property is enforced by trusted hardware or software [24, 38, 39]; that is, in practice, this assumption is widely adopted by publishers (to whom it matters most). Moreover, returning soundness is *impossible* to enforce classically without trusting some mechanism for deletion and preventing copying before deletion. Even in physical lending, perfect complete copy prevention is not possible; indeed, this fact motivates our definition of copy deterrence, an attempt to model the practical difficulty-but-not-impossibility of creating copies of physical books. ***Copy deterrence*** is also enforced by trusted hardware or software in current systems, and is similarly impossible digitally without some trust assumption deterring copying of information.

In sum, we selectively invoke trusted hardware only for properties: (1) where trusted hardware or software is already widely accepted and relied upon in practice; (2) impossible without trust from some hardware or software; and (3) that concern *access control*, in which limited context we treat digital licensing norms as an acceptable fallback standard (see § 4).

Library Catalog Falsification. The possibility of falsifying logs is inherent to any logging system; no measure can guarantee logs correspond to real-world events. Our soundness definitions ensure only that the library's *logged* actions conform to auditors' rules (such as loaning out only one copy of a book at a time). If the library catalog may be falsified, what is the point of LendLocked?

Libraries have always, in principle, had the ability to make and distribute illegitimate (physical or digital) copies of books without listing them in their catalogs. In practice, if a library handed out scanned piles of printer paper at scale, this would be easily detectable by employees and patrons who could blow the whistle. Our best attempt at replicating this intuitive property in the digital setting is to enable e-readers and users to detect and report any library misconduct where their borrowed materials are not reflected in the catalog (see "Whistleblowing" in § 5.2).

In general, having trusted hardware devices take actions without user cooperation may be problematic. We stress that here, the device would not reveal any *user* data, but would flag only *library* misconduct, only when *borrowed materials are detected to be missing from the catalog*, and we would recommend that automatic reporting be a device setting that can be turned on or off by users. We mention the possibility of automatic reporting to push back against the influential narrative that library-managed digital lending solutions are inherently unacceptable due to excessive risks of illegal overloaning (see §4). We believe that a system like LendLocked would better serve libraries' and patrons' interests than the status quo, and we aim to demonstrate a broad design space for constraining overloaning beyond what incumbents might suggest.

5.4 Revisiting Our Desiderata

We return to our initial desiderata from Section 4—namely, preserving the benefits in Theme 1, mitigating the concerns in Themes 2–5, and ensuring access control comparable to existing digital licensing systems—and explain briefly how each has influenced our model.

On *Themes 1–2* (benefits of digital lending and lack of library control), our model enables libraries to set up and manage their own

[8] Physical libraries also sometimes have cards in individual books that reveal the past several lends of that book, *including patron information*. Thus, in some ways, physical libraries can leak *more* than our model permits.

[9] This generalizes straightforwardly to checking that there are only n active loans on a given title at any given time, for situations where that is permissible.

[10] Indeed, users are not typically modeled separately from their device. The present discussion also illuminates why we diverge from this modeling norm.

digital lending system for any digital materials they hold and have the rights to distribute, without relying on and paying for publishers and third-party intermediary who control digital lending systems. If they deploy our system, it can run in parallel with their existing physical and digital lending systems, serving complementary needs. Addressing *Themes 3–4* (opacity, integrity, and preservation), our design is publicly available and our model is designed for libraries to manage their content and data flows in-house. Thus, we eliminate frustrations from libraries' lack of visibility into how third-party platforms are handling libraries' data, and restore library autonomy over integrity checks and long-term preservation strategies for the digital content that they manage. On *Theme 5* (surveillance and privacy), our privacy definitions bring privacy in digital lending much closer to the standard of physical lending (and reading), by eliminating the flow of patron data through third-party service providers (apart from publicly accessible catalog information) — meaning that any patron data flows stay between the library and the patron, in keeping with centuries of library tradition — and by keeping as many operations as possible local on the e-reader, to minimize patron data flows to the library. Our construction demonstrates such privacy protection is possible and practical.

Finally, through our novel auditing and copy deterrence requirements, our model achieves the same standards of *access control* as existing digital (licensing) solutions widely accepted by publishers and distributors (based on DRM via hardware security), and enables novel kinds of access control checks, e.g., ensuring compliance with limits on the number of copies that can be loaned out at a time.

Our modeling and construction do *not* address *all* concerns in our thematic analysis. Since some concerns cannot be addressed by purely technical means, we leave them as future directions.

5.5 Privacy Comparison With Physical Lending

Our model captures *incomparable* privacy properties to physical library lending, our motivating baseline in § 4: a rigorous comparison is impossible as no computational notion can capture analogue attacks. That said, we offer some comparative discussion here.

The catalog logBB in our model may publish *which books were available or loaned out at what times*. In our construction, any observer can access this information in a straightforward and scalable manner. Most physical lending settings (arguably) involve more friction to obtain the same information. For libraries doing physical lending with online catalogs, an observer can scrape the catalog to enumerate each copy's availability. The logBB in our model exposes the same information as exposed by such scraping.[11] For libraries without digital catalogs, an analogue way to reconstruct the loan information is to physically observe the library patrons and shelves, which requires substantial human effort. Given the prevalence of digital catalogs in physical lending libraries, we treat this loan information exposed in the catalog as acceptable, and instead largely focus on protecting *linkage* between patrons and loans.

The incomparability of analogue and computational notions also shows up in our reading privacy definition (Def. 13). Our definition cannot capture analogue attacks, e.g., where a physical adversarial looks over ones' shoulder to monitor a patron's page turns.

[11] But we note that our model permits leakage of unique book identifiers (identifying distinct copies of the same book), which may not always be present in online catalogs.

6 Construction of LendLocked

Our construction is in the random oracle model with trusted setup. We begin by introducing building block primitives.

A ***Key Derivation Function (KDF)*** takes as input a security parameter λ and a secret s, and outputs a pair of cryptographic keys: $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KDF}(1^\lambda, s)$ [37].

A ***signature scheme*** is a tuple $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ [35]:

- $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ The key-generation algorithm takes a security parameter λ and outputs a key-pair (vk,sk).
- $\sigma \leftarrow \mathsf{SigSign}(\mathsf{sk}, \mathsf{m})$ The signing algorithm takes as input the signing key sk, a message m, and outputs a signature σ.
- $\top/\bot \leftarrow \mathsf{SigVeri}(\mathsf{vk}, \mathsf{m}, \sigma)$ The verification algorithm takes as input the verification key vk, a message m, and a signature σ. It outputs $\top$ if σ is a valid signature on the message m under the verification key vk, and $\bot$ otherwise.

We require a signature scheme satisfying standard existential unforgeability [27] and *collision resistance* (CR). Informally, CR means that given $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, finding $\mathsf{vk}' \neq \mathsf{vk}$ such that signatures generated w.r.t. vk verify w.r.t. vk' is computationally hard. § F formally defines CR and discusses why we need it. A CR signature scheme can be easily built from a standard existentially unforgeable signature scheme and a CRHF, as detailed in § F.

A ***non-interactive zero-knowledge proof of knowledge (NIZK)***[12] is a tuple $(\mathsf{Setup}, \mathsf{NIZKProve}, \mathsf{NIZKVeri})$ [12]:

- $s \leftarrow \mathsf{Setup}(1^\lambda)$ The setup algorithm outputs a string s.
- $\pi \leftarrow \mathsf{NIZKProve}(s, x, w)$ The proving algorithm takes in s, instance x, and witness w and outputs proof π.
- $\{0, 1\} \leftarrow \mathsf{NIZKVeri}(s, x, \pi)$ The verifying algorithm either accepts the proof (outputting 1) or not (outputting 0).

We require standard completeness and non-adaptive soundness of our NIZK. Our system requires trusted setup both for the NIZK and parameter generation for, e.g., the signature scheme. In reality, a neutral third party can run the trusted setup and distribute parameters, for example one of the national library associations.

A ***commitment scheme*** (Gen, H) [35] is a tuple:

- $pp \leftarrow \mathsf{Gen}(1^\lambda)$ The setup algorithm outputs public parameters pp.
- $c \leftarrow H(pp, m)$ The randomized commitment algorithm takes pp and a message m and outputs a commitment c.

We require a commitment scheme with computational hiding and binding, and we assume all entities have access to some public parameters pp generated in an initial setup phase. For brevity, we omit pp from the input to H. It is a standard result that the binding property of H implies collision resistance. We refer to § H for standard definitions of hiding, binding, and collision resistance.

Finally, a ***Merkle-Tree-based commitment scheme*** consists of the following algorithms, and can be used to construct an *append-only log* [2, 17, 46]:

- com ← Commit(record), com = (root, len(records), aux)
- $\top/\bot \leftarrow$ CommitVeri(com, record)
- com′ ← Append(newrec, com)
- $\pi \leftarrow$ AppendProve(com, com', record)

[12] "NIZK" is often used as an abbreviation for non-interactive zero-knowledge proofs without the knowledge requirement, but we write "NIZK" to denote a non-interactive zero-knowledge proof of knowledge for brevity.

- $\top/\bot \leftarrow$ AppendVeri(com, com', π)

Based on a collision-resistant hash function, a Merkle-tree-based commitment is computationally binding. We only require binding from the Merkle commitment, while from H (above) we require both hiding and binding (for auditing privacy and soundness, respectively). In practice, the same commitment scheme satisfying both properties may be used for both H and the Merkle commitment.

Trusted Setups. In the standard model, the NIZK, the commitment H, and the Merkle commitment require a trusted setup. All three can be instantiated without trusted setup in the random oracle (RO) model. Our model and proofs implicitly assume the RO model for the NIZK and H, and thus omit trusted setup steps. We assume the Merkle tree is implemented with a Pedersen hash and trusted setup run by the library as libraries are already fully trusted to maintain their own lists of registered users, embodied in the Merkle tree. (See also "Library policies" under § 5.) Note that we use the NIZK to prove statements about the Merkle tree, another reason we prefer to avoid the use of RO in the Merkle commitment.

6.1 Protocol

Our construction LendLocked contains five protocols. Due to space, pseudocode for *Register*, *Borrow*, *Access*, *Return*, and *Audit* are in § I.4.

Register^{URL}. A patron must register as a user $\mathcal{U}$ with an e-reader $\mathcal{R}$ in $\mathcal{L}$ to use the lending service.

(a) uid, $\mathsf{st_L}$.ulst $\leftarrow$ PatronRegL(u.info, uauth) $\mathcal{L}$ generates a unique uid using the patron information u.info with authentication uauth, updates $\mathsf{st_L}$.ulst, and sends (uid) to $\mathcal{U}$.

(b) regmsg, $\mathsf{st_U}$.auth $\leftarrow$ GenRegReqU(uid, $\mathsf{st_U}$.auth) $\mathcal{U}$ stores the identifier uid received from $\mathcal{L}$ in $\mathsf{st_U}$.auth. $\mathcal{U}$ sends a registration message regmsg = (u.s, uid, uauth) to $\mathcal{R}$.

(c) regreq, $\mathsf{st_R}$.key $\leftarrow$ KeyGenRegR(1^λ, regmsg, $\mathsf{st_R}$.key) $\mathcal{R}$ first checks that there is no existing keys in $\mathsf{st_R}$.key to avoid double registration. Then, $\mathcal{R}$ generates a key pair ($\mathsf{vk_{ur}}$, $\mathsf{sk_{ur}}$) using the user secret u.s and the e-reader side secret $\mathsf{st_R}$.s. $\mathcal{R}$ stores ($\mathsf{vk_{ur}}$, $\mathsf{sk_{ur}}$) in $\mathsf{st_R}$.key and outputs regreq $\leftarrow$ (uid, uauth, $\mathsf{vk_{ur}}$) to $\mathcal{L}$.

(d) regres, $\mathsf{st_L}$.reglst, logBB $\leftarrow$ ProcessRegReqL(regreq, $\mathsf{st_L}$.reglst, logBB) $\mathcal{L}$ parses regreq, updates its internal database of registered $\mathcal{R}$ from $\mathcal{U}$ by adding (uid, $\mathsf{vk_{ur}}$) into $\mathsf{st_L}$.reglst. $\mathcal{L}$ adds $\mathsf{vk_{ur}}$ as a leaf node in a Merkle tree containing all registered $\mathcal{U}$-$\mathcal{R}$ verification keys, and appends ($\mathsf{vk_{ur}}$, com_{vk}') to logBB. $\mathcal{L}$ sends a reference regres = vkref of this appending to $\mathcal{R}$.

(e) $\top/\bot$, $\mathsf{st_R}$.key $\leftarrow$ FinishRegR(regres, $\mathsf{st_R}$.key) $\mathcal{R}$ processes the input regres and verifies the correctness of the just committed and appended $\mathsf{vk_{ur}}$. In case verification fails, $\mathcal{R}$ aborts and clean the state in $\mathsf{st_R}$.key.

Borrow^{URL}. After registration, a user $\mathcal{U}$ with an e-reader $\mathcal{R}$ can run the borrow protocol together with $\mathcal{L}$ to loan out a book b.title.

(a) *(Optional)* holdloan $\leftarrow$ PlaceLoanU(b.title, $\mathsf{st_U}$.auth) $\mathcal{U}$ can optionally place a loan to $\mathcal{L}$ to hold the book without involving $\mathcal{R}$, i.e., $\mathcal{U}$ takes as input the book b.title and the user authorization information $\mathsf{st_U}$.auth. $\mathcal{U}$ sends the hold request of loaning b.title to $\mathcal{L}$. We omit this optional procedure in Algorithm 2.

(b) loanmsg, $\mathsf{st_U}$.eph $\leftarrow$ GenLoanMsgU(1^λ, b.title, $\mathsf{st_U}$.auth) To borrow b.title, $\mathcal{U}$ generates a pair of ephemeral keys (epk, esk) to form a loan request loanmsg. $\mathcal{U}$ sends loanmsg to $\mathcal{R}$ and stores $\mathsf{st_U}$.eph = (b.title, ts, epk, esk, σ_{eph}).

(c) loanreq, $\mathsf{st_R}$.req $\leftarrow$ GenLoanReqR(loanmsg, $\mathsf{st_R}$.key) $\mathcal{R}$ processes the input loanmsg = (b.title, ts, epk, σ_{eph}, uid, uauth). If σ_{eph} verifies, $\mathcal{R}$ uses $\mathsf{st_R}$.key to generate a Non-Interactive Zero-Knowledge Proof (NIZK) π_{loan} which proves that the loan is requested by a registered $\mathcal{U}$-$\mathcal{R}$ pair on the committed and appended Merkle tree, without revealing the exact identity indicated by $\mathsf{vk_{ur}}$. They also commit σ_{ur} and σ_{eph} via H, storing the commitment randomness for the commitment until the return of that loaned item is proccessed. This additional commitment via H is needed to ensure that no two borrow/return records can be parsed by e-readers as being for distinct users (see auditing soundness); and we need hiding here (unlike in the Merkle commitment) for patron anonymity because σ_{ur} may reveal patron identity (see auditing privacy). $\mathcal{R}$ sends loanreq to $\mathcal{L}$.

(d) loanres, logBB, $\mathsf{st_L}$.loan $\leftarrow$ ProcessLoanReqL(loanreq, logBB, $\mathsf{st_L}$.rul) $\mathcal{L}$ parses loanreq from $\mathcal{U}$ and verifies:
  i) $\mathcal{U}$ is an authorized patron
  ii) b.title has at least one available copy
  iii) this loan complies with lending rules, e.g., age limit, upper bound of open loans per user etc.
  iv) σ_{eph} and π_{loan} verifies
  Upon verifications, $\mathcal{L}$ approves this loan request by adding (b.title, uid, ddl) in the open loans database $\mathsf{st_L}$.loan. $\mathcal{L}$ then computes and appends a record on logBB as (b.title, ts, epk, σ_{eph}, π_{loan}, c'_{BB}, σ_{sk_L}, π_{apnd}) where c_{BB}' is the updated commitment appended (b.title, ts, epk, σ_{eph}, π_{loan}); σ_{sk_L} provide non-refutability of $\mathcal{L}$ adding this record on logBB; π_{apnd} which is an append-only proof for auditing. Finally, $\mathcal{L}$ sends loanres = (logref, ddl) to $\mathcal{R}$ where logref refers to the line number on logBB to facilitate the verification of the just-appended record in the next step, ddl is the return deadline for this loan.

(e) $\mathsf{st_R}$.ctdb $\leftarrow$ VerLoanResR(loanres, logBB, $\mathsf{st_R}$.key, $\mathsf{st_R}$.req) $\mathcal{R}$ parses loanres, verifies σ_{sk_L} is valid, the NIZK verifies according to the information in $\mathsf{st_R}$.req, and logref is consistent with the values in $\mathsf{st_R}$.req via self-auditing in Algorithm 5. If not, $\mathcal{U}$ and $\mathcal{R}$ abort. Otherwise, $\mathcal{R}$ associates ddl with loan b.title in $\mathsf{st_R}$.ctdb for potential enforcement of return and deletion of b.title content in the return protocol.

Access^{URL}. After successfully executing the borrow protocol on b.title, $\mathcal{U}$ using $\mathcal{R}$ can fetch the content b.cont of the book b.title from $\mathcal{L}$, and access b.cont as wish before the return deadline ddl.

(a) accessreq $\leftarrow$ PrepAccessU(b.title, $\mathsf{st_U}$.auth, $\mathsf{st_U}$.eph) $\mathcal{U}$ sends an access request accessreq to $\mathcal{R}$ accessreq $\leftarrow$ (b.title, $\mathsf{st_U}$.auth.uid, $\mathsf{st_U}$.auth.uauth, $\mathsf{st_U}$.eph.esk).

(b) loadreq $\leftarrow$ GenLoadReqR(accessreq, logBB, $\mathsf{st_R}$.ctdb, $\mathsf{st_R}$.key) $\mathcal{R}$ parses accessreq as (b.title, uid, uauth, esk). In case ddl passed, $\mathcal{R}$ removes any previously downloaded content b.cont from $\mathsf{st_R}$.ctdb and invokes GenRetReqR() of

Return^{URL} with esk as part of the input. Otherwise, $\mathcal{R}$ assures the borrow record is still open on logBB, and there is no return record afterwards. If b.title was downloaded, $\mathcal{R}$ securely present b.cont to $\mathcal{U}$ by running the representations function repr. Otherwise, $\mathcal{R}$ sends loadreq as (logref, uid, uauth, $\mathsf{vk}_{\mathsf{ur}}$) to $\mathcal{L}$.

(c) loadres ← ProcessLoadReqL(loadreq, logBB, st_{L}.reglst, st_{L}.ulst, st_{L}.loan) $\mathcal{L}$ parses loadreq, and verifies:
  i) (uid, uauth) ∈ st_{L}.ulst, (uid, $\mathsf{vk}_{\mathsf{ur}}$) ∈ st_{L}.reglst
  ii) ∃(b.title, uid, ddl, ⊥) ∈ st_{L}.loan,
  iii) $C(\cdot)$ < ddl, the return deadline ddl has not passed
  Upon verifications, $\mathcal{L}$ sends b.cont to $\mathcal{R}$ and updates the loan status as loaded (b.title, uid, ddl, ⊤) in st_{L}.loan.

(d) accessres ← ProcessLoadResR(loadres, st_{R}.ctdb) $\mathcal{R}$ stores the content b.cont in st_{R}.ctdb and runs repr(b.cont) to securely present b.cont to $\mathcal{U}$.

Return^{URL}. $\mathcal{U}$ can run the return protocol to terminate its access to b.title before the return deadline ddl. If not, upon an access request passing ddl, $\mathcal{R}$ runs the return protocol. The return protocol ensures deletion of the book content b.cont from $\mathcal{R}$ and publish a return record on the catalog logBB to close the corresponding open loan.

(a) rmreq ← PrepRetU(b.title, st_{U}.eph, st_{U}.auth) $\mathcal{U}$ returns b.title by sending $\mathcal{R}$ an unloading request rmreq ← (b.title, st_{U}.eph.esk, st_{U}.auth.uid, st_{U}.auth.uauth).

(b) retreq ← GenRetReqR(rmreq, st_{R}.ctdb, st_{R}.key)
Either upon receiving rmreq before the return deadline ddl from $\mathcal{U}$, or $\mathcal{R}$ goes online for the first time passing ddl, $\mathcal{R}$ sends $\mathcal{L}$ a return request retreq ← (b.title, $C(\cdot)$, esk, st_{R}.ctdb[b.title].logref, uid, uauth, st_{R}.key.$\mathsf{vk}_{\mathsf{ur}}$)

(c) retres, logBB, st_{L}.loan ← ProcessRetReqL(retreq, logBB, st_{L}.loan) $\mathcal{L}$ parses retreq as (b.title, ts, esk, ptr, uid, uauth, $\mathsf{vk}_{\mathsf{ur}}$) and verifies:
  i) (uid, uauth) ∈ st_{L}.ulst, $\mathcal{U}$ is a registered patron
  ii) (uid, $\mathsf{vk}_{\mathsf{ur}}$) ∈ st_{L}.reglst, $\mathsf{vk}_{\mathsf{ur}}$ is registered under uid
  iii) ∃(b.title, uid, ddl, ∗) ∈ st_{L}.loan, $\mathcal{U}$ with uid has an open loan on b.title within the return deadline ddl
  iv) SigVeri(esk, logBB [logref.σ_{eph}], logBB [ptr]) == ⊤, esk is the corresponding signing key of epk in the record denoted by logref on logBB
  After verifications, $\mathcal{L}$ removes (b.title, uid, ddl) from the internal open loan database in st_{L}, computes, and appends a record (b.title, ts = min(ts, ddl), esk, ptr, c_{BB}, σ_{sk_L}, π_{apnd}, $h = H(\sigma_{\mathsf{ur}}, \sigma_{\mathsf{eph}})$) on logBB. [13]
  (a) st_{L}.loan ← st_{L}.loan − (b.title, uid, ddl, ∗)
  (b) c_{BB} ← Commit(b.title || ts || esk || ptr)
  (c) c'_{BB} ← Append(b.title || ts || esk || ptr, c_{BB})
  (d) π_{apnd} ← AppendProve(c_{BB}, c_{BB}', b.title || ts || esk || ptr)
  (e) σ_{sk_L} ← SigSign(sk_L, b.title || ts || esk || ptr || c_{BB}')
  (f) logBB ← logBB + (b.title, ts, esk, ptr, c_{BB}, σ_{sk_L}, π_{apnd})
  (g) retres ← #logBB[(b.title, ts, esk, ptr, c_{BB}, σ_{sk_L}, π_{apnd})] where # denotes the reference, e.g., a line number.
  To confirm a successful return, $\mathcal{L}$ sends retres back to $\mathcal{R}$.

(d) retack, st_{R}.ctdb ← ProcessRetResR(retres, st_{R}.ctdb) $\mathcal{R}$ verifies the return record using the reference location in retres.
  Upon verification, $\mathcal{R}$ deletes the local cache of (b.title, b.cont) in st_{R}.ctdb, then sends an acknowledgement retack = ⊤ to $\mathcal{U}$. Otherwise, $\mathcal{R}$ sends $\mathcal{U}$ retack = ⊥ to notify the user of a failure and aborts from the protocol.

Audit^{V}. $\mathcal{V}$ can run the non-interactive audit protocol to check the library lending system via monitoring the catalog logBB.

(a) ⊤/⊥ ← EvalAuditProofV(tsrange, logBB, DistLst) $\mathcal{V}$ is free to choose the auditing period tsrange as long as it includes the time when last audit happened.
  For each records within tsrange, $\mathcal{V}$ verifies:
  i) AppendVeri(com, com', π) = ⊤, make sure logBB is append-only since the last audit;
  ii) σ_{sk_L} verifies, i.e., $\mathcal{L}$ has signed all the appended records;
  iii) SigVeri(logBB [ptr].epk, logBB [ptr].σ_{eph}, esk) == ⊤, each return record has the esk that is the corresponding signing key of the epk inside the pointed borrow record;
  iv) No borrow record referred by over one return records;
  $\mathcal{V}$ uses the records on logBB to compute the number of loaned copies of b.title in tsrange and cross-checks with distribution list DistLst. If the number of simultaneous open loans of any book is no greater than as defined in DistLst, $\mathcal{V}$ returns ⊤. Otherwise, it returns ⊥ meaning $\mathcal{L}$ has loaned more copies for at least one book than defined by DistLst.

7 Proving Security

Theorem 1. *LendLocked (Section 6) is a private, transparent digital lending system (Definition 1), when it is instantiated with a secure KDF, signature scheme, NIZK, commitment scheme, and collision-resistant hash function, and with a secure TEE $\mathcal{R}$.*[14]

Our proof proceeds modularly: Theorem 1 follows from Theorems 2 and 4, which state LendLocked adheres to each security property in § 5.2. Due to space, all proofs are in § J.

8 Evaluation by Micro-Benchmarks

We implemented the key cryptographic operations in LendLocked in Python, and micro-benchmarked their performance. We micro-benchmark Borrow and Audit only, as these are where we introduce overhead compared to existing systems. We did not implement our full protocols, just cryptographic operations, as these capture our overhead. For the speed of standard cryptographic operations we also refer to prior literature. We omit any evaluation of protocols Register, Load, and Return, as LendLocked doesn't add overhead on these compared to existing digital lending systems.

Our measurements were run on a MacBook with Apple M1 Pro chip 1.30GHz and 16 GB memory.

Borrow Protocol. We choose Edwards-curve Digital Signature Algorithm (EdDSA) [53], and use the Pedersen hash[15] to construct the Merkle tree for the membership proof π_{mem} in the loan proof π_{loan} [61].[16] We choose Pedersen over SHA-256 as it is about 5× faster with half of the memory usage in proof generation [59]. The

[13] For enforced return, the return time is set as ddl, since e-reader will delete the content and remove the possibility to access after it comes online for the first time.

[14] The CRHF is needed for the Merkle commitment, not used directly in the construction. See Section 6 for definitions of these cryptographic primitives and the security properties that LendLocked requires from each. See Section 5 for our TEE assumption.

[15] Which is a collision-resistant hash function as required in §6.

[16] § F provides brief discussion on signature collision resistance and EdDSA.

Table 3: Measurements of Proof Functions.

Computed By		Setup $\mathcal{R}$	Witness $\mathcal{R}$	Prove $\mathcal{R}$	Verify $\mathcal{L}$
Online		No	Yes	Yes	Yes
Merkle Depth		Time (s)			
10	Mean	2.375	1.414	2.207	0.006
	Stddev	0.025	0.021	0.116	0.002
	Median	2.363	1.405	2.163	0.005
20	Mean	2.794	1.570	2.447	0.006
	Stddev	0.063	0.012	0.012	0.001
	Median	2.806	1.569	2.444	0.005
30	Mean	3.104	1.743	2.757	0.005
	Stddev	0.013	0.015	0.011	0.001
	Median	3.109	1.740	2.754	0.005

efficiency of proof generation for π_{loan} is vital, as it happen online on the e-reader; other operations like Setup and Verify happen either offline or are done by the library.

We instantiate our NIZKs with zkSNARKs [11] using the library ZoKrates [60]. We considered other zkSNARK tools like Circom [20] and Arkworks [8], and chose ZoKrates for its python support libraries [5, 62] for the cryptographic primitives we need.

Table 3 shows the time measurement of different computations regarding the loan proof π_{loan}. The proof of possessing a valid signature signed by $\mathsf{sk}_{\mathsf{ur}}$ does not change with the scale of registered $\mathcal{U}$-$\mathcal{R}$ pairs, while the proof on the possession of a valid membership proof π_{mem} does scale up. We measure on the scale of 1K, 1M, and 1B registered $\mathcal{U}$-$\mathcal{R}$ pairs; this covers even the largest libraries [57].

As $\mathcal{R}$ can compute Setup offline, Setup does not impact online user experience. The largest overhead is in borrowing, where the largest computation time at the scale of a billion registered $\mathcal{U}$-$\mathcal{R}$ pairs is within five seconds.[17] While this is noticeably slower than current digital book licensing apps, we believe it is tolerable as a one-time cost per borrowed book that does not affect reading experience or returns once a book is borrowed. (It is also considerably faster than checking out a physical book from a library.) Improving this cost, perhaps leveraging future zkSNARK advances or the fact that e-readers in LendLocked perform many proofs of a very similar format, is a clear area for improvement on LendLocked's efficiency. Proof verification time is a few milliseconds.

Audit Protocol. Our auditing, in computations and scale, is similar to those in the literature and practice of transparency logs [21, 23, 40, 55]. For each record, there are at most three cryptographic operations in the audit protocol Audit^V: two signature verifications and verifying the append-only proof π_{apnd}. Signature verification is constant, e.g., verifying an EdDSA signature takes about 0.971 ms [53]. The overhead of verifying the append-only proof depends on the scale of newly-added records. Monthly circulations at the largest libraries worldwide are around 50M [49], which leads to 100M catalog records appended per month; this is a comfortable overestimate as these figures include physical circulation. With monthly audits, each append-only proof π_{apnd} would contain 27 hashes. Recomputing all hashes (0.015 ms for 30 hashes) and verifying the two signatures (1.942 ms) to validate π_{apnd} would take about 1.957 ms. In total, verifying 100M records takes 2 to 3 days on a MacBook. Multi-day background computations for continuous auditing are a real-world practice (Appendix D.5), so performing Audit^V monthly is practical.

EdDSA and Pedersen Hashes. Note that EdDSA and Pedersen hashes both rely on elliptic-curve Diffie-Hellman, so we must take care about parameter cross-dependencies. Both will be secure provided the generator point for EdDSA and the generator points for the Pedersen hash are chosen randomly and independently.

[17] This scale is a comfortable over-estimate as the largest libraries in the world have a few million visitors per year [57].

9 Extensions

We refer to Appendix C for discussion on supporting multiple e-readers per user account, achieving formal notions of privacy for licensing, and notions of privacy against surveilling libraries.

10 Conclusion

We conduct an "end to end" exploration of *privacy and transparency in digital lending* from expert interviews to establish context and needs, to modeling, design, and evaluation. Our thematic analysis documents key concerns and needs related to library technologies, including but not limited to privacy and transparency, highlighting a range of open directions for future work, from privacy technologies research to socio-technical and economic issues. Informed by our interviews, we provide the *first rigorous modeling and construction* of private and transparent digital library lending, Our system design, LendLocked, is based on lightweight cryptographic components, provably satisfies rigorous privacy, transparency, and access control guarantees, and has tolerable efficiency at the scale of the largest libraries. Our work offers a foundation for the development of open, privacy-respecting digital lending tools, a domain with pressing needs thus far underexplored by our research community.

Acknowledgments

We are grateful to our interviewees for their generosity with their time, and for sharing their perspectives. We thank Jason Schultz, Rosanna Bellini, and Rachel Greenstadt for helpful discussions, and Stephanie Chen and Naveen Rajan for research assistance. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

References

[1] 2006. *DRM '06: Proceedings of the ACM workshop on Digital rights management.* Association for Computing Machinery. https://doi.org/10.1145/1179509
[2] Adam Eijdenberg, Ben Laurie, Al Cutter. 2015. Verifiable Data Structures. https://perma.cc/A732-TT3N. Accessed: 2025-04-10.
[3] Kendra Albert and James Grimmelmann. 2023. Do the Right Thing. *Commun. ACM* (2023). https://doi.org/10.1145/3587825
[4] Authors Alliance. 2021. Libraries, COVID-19, and E-Book Lending: One Year Later. https://perma.cc/U7L5-ZTN3. Accessed: 2025-04-14.
[5] Alvaro Alonso. 2024. ZnaKes. https://perma.cc/VFG5-XA5V. Accessed: 2025-04-08.
[6] American Library Association. 2023. New ALA Report Maps Increasingly Complex Digital Public Library Ecosystem. https://perma.cc/K4WK-DELR. Accessed: 2024-04-27.
[7] Nate Anderson. 2006. Hacking Digital Rights Management. https://perma.cc/PYW3-TSGW.

[8] Arkworks Github. 2022. Arkworks Implementation of Groth16. https://perma.cc/NQY2-U8X8. Accessed: 2024-05-09.

[9] Article 19. 2026. Iran: Protests continue as repression of free expression intensifies. https://perma.cc/V2JD-PADC.

[10] American Library Association. 2003. Resolution on the USA Patriot Act and Related Measures that Infringe on the Rights of Library Users. https://perma.cc/5LKF-S5B8.

[11] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. 2012. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *ITCS*. https://doi.org/10.1145/2090236.2090263

[12] Manuel Blum, Paul Feldman, and Silvio Micali. 1988. Non-Interactive Zero-Knowledge and Its Applications. In *ACM Symposium on Theory of Computing - STOC 1988*. https://doi.org/10.1145/3335741.3335757

[13] Bonnie Tijerina Bobbi Newman. 2017. *Protecting Patron Privacy: A LITA Guide*. Rowman and Littlefield. https://doi.org/10.1080/1941126X.2018.1443990

[14] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* (2006). https://doi.org/10.1191/1478088706qp063oa

[15] Jan Camenisch and Anna Lysyanskaya. 2001. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Eurocrypt*. https://doi.org/10.1007/3-540-44987-6_7

[16] PEN American Center. 2023. PEN Welcomes UN Action on Right to Privacy. https://perma.cc/N3FS-NWAC. Accessed: 2025-04-14.

[17] Melissa Chase and Sarah Meiklejohn. 2016. Transparency Overlays and Applications. In *CCS*. https://doi.org/10.1145/2976749.2978404

[18] Benny Chor, Amos Fiat, and Moni Naor. 1994. Tracing Traitors. In *CRYPTO*. https://doi.org/10.1007/3-540-48658-5_25

[19] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. 1995. Private Information Retrieval. In *36th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. https://doi.org/10.1109/SFCS.1995.492461

[20] Circom Github. 2021. Circom Compiler For ZK Proof Systems. https://perma.cc/23DV-Y9NV. Accessed: 2025-04-11.

[21] Cloudflare. 2025. A next-generation Certificate Transparency log built on Cloudflare Workers. https://perma.cc/HUV9-EQ82. Accessed: 2025-04-11.

[22] Danielle Cooper. 2021. Licensing Privacy: Views From Library Leadership. https://perma.cc/52AH-JA9B. Accessed: 2025-04-14.

[23] Scott A. Crosby and Dan S. Wallach. 2009. Efficient Data Structures For Tamper-Evident Logging. In *Usenix Security*. https://perma.cc/LHE9-DTHZ

[24] Greg Farough. 2023. Worldwide community of activists protest OverDrive and others forcing DRM upon libraries. Free Software Foundation. https://perma.cc/G9VX-WZKQ.

[25] Federal Trade Commission. 2024. A Look Behind the Screens: Examining the Data Practices of Social Media and Video Streaming Services. FTC Staff Report. https://perma.cc/HB54-5G6M.

[26] Loida Garcia-Febo, Anne Hustad, Hermann Rösch, Paul Sturges, and Amelie Vallotton. 2012. IFLA Code of Ethics for Librarians and Other Information Workers. https://perma.cc/U7H7-VXR2.

[27] Shafi Goldwasser, Silvio Micali, and Ronald Rivest. 1988. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. In *SIAM Journal of Computing*. https://doi.org/10.1137/0217017

[28] Michael Gorman. 2015. Our Enduring Values Revisited: Librarianship in an Ever-Changing World. https://perma.cc/TRL9-E2EH.

[29] Joint Digital Content Working Group. 2020. The Need for Change: A Position Paper on E-Lending. https://perma.cc/6ZHG-DUSJ. Accessed: 2025-04-14.

[30] David R. Hansen and Kyle K. Courtney. 2018. A White Paper on Controlled Digital Lending of Library Books. https://perma.cc/9EHA-4PPX. Accessed: 2024-04-26.

[31] Joacim Hansson. 2016. The documentality of ethics: Codes of library ethics as support of professional practice. https://perma.cc/NG8D-BNET. In *Proceedings from the Document Academy*.

[32] Nate Hoffelder. 2015. ALA Releases New Digital Privacy Guidelines. https://perma.cc/P2YV-YEFC. Accessed: 2025-04-14.

[33] International Organization for Standardization (ISO). 2019. Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems. https://perma.cc/KWC7-J5EM ISO 9241-210:2019.

[34] Seny Kamara. 2022. Crypto for the People (part 2). USENIX ENIGMA, Santa Clara, CA. https://perma.cc/2X2N-N43W

[35] Jonathan Katz and Yehuda Lindell. 2020. Introduction to Modern Cryptography. https://perma.cc/763U-C49L

[36] Anne Klinefelter. 2007. Privacy and Library Public Services: Or, I Know What You Read Last Summer. *Legal Reference Services Quarterly* (2007). https://doi.org/10.1300/J113v26n01_13

[37] Hugo Krawczyk. 2010. Cryptographic Extraction and Key Derivation: The HKDF Scheme. In *CRYPTO*. https://doi.org/10.1007/978-3-642-14623-7_34

[38] Sarah Lamdan, Jason M. Schultz, Michael Weinberg, and Claire Woodcock. 2023. The Anti-Ownership Ebook Economy. Engelberg Center on Innovation Law and Policy at New York University. https://perma.cc/UCU2-UFAC.

[39] C.A. Lemmer and C.P. Wale. 2016. *Digital Rights Management: The Librarian's Guide*. Bloomsbury Publishing. https://perma.cc/P4H5-NZGP

[40] Let's Encrypt. 2023. Certificate Transparency (CT) Logs. https://perma.cc/8P2H-8L4R. Accessed: 2025-04-11.

[41] Wei-Kai Lin, Ethan Mook, and Daniel Wichs. 2023. Doubly Efficient Private Information Retrieval and Fully Homomorphic RAM Computation from Ring LWE. In *55th Annual ACM Symposium on Theory of Computing STOC*, Barna Saha and Rocco A. Servedio (Eds.). ACM. https://doi.org/10.1145/3564246.3585175

[42] Alan F. Luo, Noel Warford, Samuel Dooley, Rachel Greenstadt, Michelle L. Mazurek, and Nora McDonald. 2023. How Library IT Staff Navigate Privacy and Security Challenges and Responsibilities. In *Usenix Security*. https://perma.cc/PQ9J-D3HJ

[43] Emily Zoe Mann, Stephanie A. Jacobs, Kirsten M. Kinsley, and Laura I. Spears. 2023. Tracking transparency: an exploratory review of Florida academic library privacy policies. *Information and Learning Sciences* (2023). https://doi.org/10.1108/ILS-04-2023-0038

[44] Rod McCullom. 2026. Inside Chicago's surveillance panopticon. MIT Technology Review. https://perma.cc/2CXQ-C494.

[45] Nora McDonald, Rachel Greenstadt, and Andrea Forte. 2023. Intersectional Thinking about PETs: A Study of Library Privacy. *PoPETS* (2023). https://doi.org/10.56553/POPETS-2023-0064

[46] Ralph C. Merkle. 1987. A Digital Signature Based on a Conventional Encryption Function. In *CRYPTO*. https://doi.org/10.1007/3-540-48184-2_32

[47] David Molnar and David Wagner. 2004. Privacy and Security in Library RFID Issues, Practices, and Architectures. In *CCS*. https://doi.org/10.1145/1030083.1030112

[48] Isaiah Munyoro. 2017. Issues of confidentiality and public interest : a case study from a library perspective. https://doi.org/10.25159/0027-2639/2413. In *Mousaion: South African Journal of Information Studies*, Vol. 35. Issue 3.

[49] New York Public Library. 2012. NYPL BY THE NUMBERS. https://perma.cc/2ZZJ-RZ6Y. Accessed: 2025-04-10.

[50] Young-Hee Noh. 2012. A Study of Digital Library Service Records and User Privacy. https://doi.org/10.3743/KOSIM.2012.29.3.187. In *Journal of the Korean Society for information Management*, Vol. 29. Issue 3.

[51] Rachel Noorda and Kathi Inman Berens. 2023. Digital Public Library Ecosystem Report. https://perma.cc/2WAY-JYTP. Accessed: 2024-04-26.

[52] Donald A. Norman. 2002. *The design of everyday things*. Basic Books. https://perma.cc/3T8X-K8KD

[53] I. Liusvaara S. Josefsson. 2017. Edwards-Curve Digital Signature Algorithm (EdDSA). https://perma.cc/PYB9-3Q9X. Accessed: 2025-04-04.

[54] Amita Sethi. 2004. Digital Rights Management and Code Obfuscation. https://perma.cc/UGV5-2AKB

[55] Alin Tomescu, Vivek Bhupatiraju, Dimitrios Papadopoulos, Charalampos Papamanthou, Nikos Triandopoulos, and Srinivas Devadas. 2019. Transparency Logs via Append-Only Authenticated Dictionaries. In *CCS*. https://doi.org/10.1145/3319535.3345652

[56] Hans H. Wellisch. 1981. Ebla: The World's Oldest Library. https://perma.cc/WXH2-VLAV.

[57] Wikipedia. 2025. List of Largest Libraries. https://perma.cc/F6F4-UUJA. Accessed: 2025-04-11.

[58] Andrew Yao. 1982. Protocols for Secure Computations. In *23rd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. https://doi.org/10.1109/SFCS.1982.38

[59] ZKPlus. 2025. Benchmarks of Hash Algorithms in ZoKrates. https://perma.cc/WZA3-ZS3P. Accessed: 2025-04-08.

[60] ZoKrates Github. 2018. ZoKrates. https://perma.cc/CP7R-WALR. Accessed: 2025-04-08.

[61] ZoKrates Github. 2021. Pedersen Hash in ZoKrates. https://perma.cc/JG84-SD53. Accessed: 2025-04-08.

[62] ZoKrates Github. 2021. ZoKrates pyCrypto. https://perma.cc/GKH7-CLB2. Accessed: 2025-04-08.

Appendices

A Ethical Considerations & Societal Impact

The interview portion of our study was approved by our institutional/ethics review board. Participants provided explicit consent to participation before their interview, and had the options to request (1) that identifying information about them be kept confidential, and (2) not to have their interview audio recorded. We respected

these choices strictly, and carefully considered possible adverse consequences of releasing identifying information also about the participants who did not explicitly request confidentiality, ultimately resulting in the decision not to release participant identities.

We do not perceive ethical dilemmas raised by the technical portion of our work (modeling, construction, and evaluation) towards stakeholders in the library ecosystem: libraries, patrons, publishers, intermediaries, and other rights-holders. (We propose a new approach in a market of digital lending solutions dominated by certain publishers/platforms; our proposal may be in tension with their business interests, as any new proposal might. This is outside the relevant scope of ethical considerations for academic research.)

We note that there has been a lengthy debate in the related context of Controlled Digital Lending (CDL) (see § 2) about whether CDL-based library lending practices are compliant with copyright law, and without getting into a tangential discussion of how exactly compliance with copyright law intersects with ethical considerations,[18] we emphasize that our proposed system is designed to facilitate transparent *compliance* with any applicable constraints under copyright law.

B Open Science

We have included the artifacts of the interview study, formal model, construction algorithms, and cryptographic proofs in the main body and the appendix of this paper. We have released the evaluation benchmarks in the Github repository: github.com/lendlocked.

C Extensions

We offer a preliminary discussion of three possible extensions to our model and construction: Supporting multiple e-readers, supporting private licensing, and supporting privacy against curious libraries.

C.1 Multiple E-readers

For simplicity, we model a single e-reader $\mathcal{R}$ per user $\mathcal{U}$ in previous sections. However, it is simple to extend our model to multiple e-readers per user. Currently, we could support a system in which a user may be independently enrolled with multiple e-readers and given the ability to borrow multiple copies of the book, one for each e-reader, in order to read it on each device. However, this is of course wasteful of the library's resources and dissimilar to how e-reading works in practice currently. Instead, in matching with our system's goal of mimicking physical lending properties, this system would treat each e-reader as a separate book that is borrowed. To extend our system to simultaneous e-reading, there are two main considerations in this process: (1) coherence between e-readers and library, and (2) limiting distribution through simultaneous accessing.

We see that our existing $\mathcal{R}$ setup already checks for new updates at fresh accesses and on startup. In theory, this ensures coherence between each e-reader and the library on the current catalog. However, our current design achieves auditing soundness by associating with each log in logBB a user-reader pair. As such, we would need to equip the readers and library with some sort of way to associate one or multiple logs across readers to a single user. This could be done using a shared key across readers, by augmenting the logs (being careful to not leak how many e-readers each user uses), or via communications between the readers.

In addition, by (1), we also mean desirable properties such as saving reading progress or in-text notes across e-readers. The natural way to achieve this would be via a communication network across a user's readers. If we can assume that each user has to enroll any new e-readers with their existing ones, this can be done easily. If not, this may still be achievable through secure multi-party computation [58], assuming a limited number of reading platforms (for efficiency's sake).

C.2 Private Licensing

As discussed in Section 4.2, licensing generally utilizes a third party to attest to access control, and many existing licensing schemes in practice are private, closed systems. For now, we will consider the case where the third party possesses the text and works with the library on sending it to users.

None of these considerations necessarily preclude a licensing scheme from meeting our model definitions. In fact, copy deterrence could in theory be more easily achieved, as the third party could send to the reader only small parts of borrowed text at a time. The downside of this would be requiring always-online access, but it is possible some hybrid between this and our existing construction could also achieve copy deterrence.

Correctness, soundness, and auditing guarantees also seem to work easily with licensing without any issue, provided the third party publishes and maintains logBB. It is even feasible that this could more easily be done on the licenser's side than the library's due to differences in technical expertise. However, it is not immediately clear how reading privacy (Definition 13 could be achieved, as the user or e-reader having to ask the third party for access whenever they wish to read immediately violates reading privacy.

Toward this, we believe private information retrieval (PIR) techniques [19] may be useful. In particular, when a reader wishes to get the next piece of text on behalf of their user, they may make a PIR query to the third party's stores of content in order to retrieve the desired piece of text. However, depending on how much of the text the reader is given at once, this may still result in some leakage about user reading patterns. In addition, it is possible the size of licensed content makes such a scheme infeasible, though recent developments in doubly efficient PIR [41] give hope to this end.

C.3 Additional Privacy Against Libraries

LendLocked already provides more privacy against libraries than current systems provide against intermediary platforms, and provides privacy against libraries commensurate to physical lending. Still, in the digital setting, we could hope to do even better. While this is not the main focus of our work, we briefly discuss some possible directions for future work here.

Similar to the question of private licensing, we may also want to consider an alternative scheme where the borrowing and returning patterns of individual users is hidden not only from outside auditors but also from the library. In this case, the library must still be able to verify which books are lent out at any given time (as otherwise it would be impossible to verify the owned-loaned ratio), but the specific identities of users and readers would be hidden.

[18] Law ≠ ethics. See also [3].

In this case, PIR techniques may also be useful here in order to hide the specific borrow requests from the library, as well as individual access requests. Combining this with an appropriate anonymous credentials may be sufficient to guarantee privacy against the library, though this may require a trusted party to issue these credentials [15]. Alternatively, it is possible the NIZKs on logBB may be able to additionally prove the borrow request is being made by someone who registered with the library in order to get private borrowing. We note that extending to this setting may require changes to the definitions or additional properties, as we did not consider this setting for our work.

Finally, it is worth considering how to achieve additional properties in this regime. For example, if a library needs to ban a user from its library or track certain borrowing habits to collect late fees, some sort of traitor tracing [18] may be desired. We leave this space as interesting future work.

D Additional Details on Interviews

D.1 Interviewee Profiles

Table 4 shows the background information of all interviewees. Participant numbers were assigned randomly. Note that all interview subjects have 10+ years experience in fields related to libraries or digital lending.

D.2 More on Theme 1 (Complementarity of digital and physical lending)

P4 mentioned that digital lending also makes different formats like audiobooks easier to use. In addition, P8 described a benefit for libraries of a predictable purchasing scheme for digital books, stating, "*We have print items that somebody spills something on, and then you have to buy it again right away.*"

Costs. Digital lending incurs ongoing costs, differently from physical lending (see also Theme 1).

> "*There have been a couple of digital services that we've been asked not to promote, because it costs per use. So . . . I don't bring it up unless [it directly relates to] the conversation that [I'm] already having [with a patron]. . . . We don't put up flyers about it.*" —P5

P5 added, "*but they might find it on our website. . . . We're discouraged from promoting it, but people still use it.*" P5 also discussed that pricing models vary, and these constraints on promotion did not apply to services which charged per time period.

As also noted under Theme 2, P11 and P1 emphasized that most libraries do not have resources to build or commission their own systems, especially public libraries, which P1 noted "*are pretty much all strapped for cash.*" P2 highlighted, relatedly, that "*the questions of equity are like—who can afford to build a new system?*"

D.3 More on Theme 2 (Lack of Control)

P5 noted that while on some services like Libby, libraries can at least purchase (limited) access to specific titles, with other services like Hoopla, *"you're just buying access to what they have. We don't have any control over that."* Similarly, P1 explained that *"publishers [can] decide to make a change in what kind of content is offered, or how that content is offered. I mean, we've even seen publishers change the contents of books and without telling people."* This kind of "*Netflix style*" (P1) lending as a service harms libraries' management of content integrity and preservation (see also Theme 4).

Power imbalance. All but three participants highlighted libraries' limited ability to change data practices, in light of the strong gatekeeper position and incentives of publishers and distributors in today's licensing landscape. Prior literature has highlighted similar concerns, e.g., [29].

P1 explained that while they have experience negotiating custom privacy-protective contract terms on behalf of library institutions, "*with these publishers and vendors . . . libraries don't always have a whole lot of negotiating power. And so, if you ask for too much, they're just going to tell you 'No.'*" P2's perspective was gloomier yet: "*I mean, it's effectively a monopoly, right? . . . The contract problem is known as probably the biggest problem right now in both public and academic libraries . . . you basically have to take whatever contract you're given.*" P2 added, *"you know, we've allowed corporate capture of so many of our public information systems."*

Funding sources. As P8 noted, some restrictions on digital offerings can also come from funding sources. They described how some grants may have restrictions on the types of digital media the money can be spent on, which can affect which offerings are purchased.

Consortia. Although consortia may have more leverage, they also reduce individual libraries' agency. P5 noted that often, technology choices and privacy policies are not made by individual libraries, but by large consortia in processes that can be "*years in the making*":

> *"So much stuff is . . . at the consortium level, that we don't have a say in or not a direct say in." —P5*

P5 once raised a privacy concern about their institution's practices with their supervisor, who agreed about the concern but expressed that not much could be done as it was "*a consortium level decision.*"

Concerns around AI.. Multiple participants expressed concerns around AI features and AI-generated content, and with libraries' lack of power and disclosure in relationships with vendors over AI features and AI-generated content (see also Theme 2). P10 and P8 voiced frustration with the proliferation of AI-generated books on digital platforms. The difficulty of removing unwanted titles that appear on digital services seems to vary from platform to platform, tying in with a lack of library control (Theme 2). P8 specifically described how Overdrive can remove specific titles by request, whereas they have had " *more issues with AI titles in Hoopla, and they are just getting uploaded right into the catalog.*" On the other hand, P10 voiced their displeasure at Overdrive rolling out an AI-powered book recommendation chatbot into the Libby app, as they were worried about bias, ineffectiveness, and errors: "*Using artificial intelligence to do that work, I find offensive.*" P6 additionally noted that it is not clear to them whether patron search terms should be used to train AI models, and more broadly what value could be extracted by platforms from users of it.

D.4 More on Theme 3 (Opacity)

P11 and P1 noted that while academic libraries may have the resources to engineer their own systems for digital lending and digital

Table 4: Interview Participants with Background Information

Participant	Role	Specializations	Main Topic Areas
P1	activist, lawyer, librarian	digital licensing patron privacy copyright law	digital lending practices
P2	activist, library scientist, library management	library association/community collection policy	digital lending practices
P3	library technologist	digital services purchasing	distributor interactions and procurement
P4	library management	administration selection	loan records
P5	librarian	patron assistance teen services	patron information
P6	library management	digital libraries digital preservation	digital lending practices
P7	library technologist	collection development digital services	distributor interactions and procurement
P8	librarian	collection development purchasing	distributor interactions and procurement
P9	activist, library management	digital content format/distribution patron privacy	patron information and loan records
P10	librarian	digital content format/distribution collection development	digital services
P11	library management	digital preservation researcher support	preservation and archival

preservation, as evidenced by a few examples, most libraries do not.

Complexity of licensing constraints. Compliance with licensing terms is a prerequisite of offering digital access, to which entire teams are dedicated. As P1 put it: "*[E]verything's just sort of this spaghetti mess of contracts that are knitted together.*" (See also Theme 3.)

D.5 More on Theme 4 (Integrity)

P1 elaborated the problems of "Netflix style" content licensing:

> "*[M]ore from a business model perspective than anything else like what the vendors really want is to get the libraries to sign up to a kind of Netflix style agreement where you get access to what they offer, but they're not always going to promise that they're offering the same things. And you know how like movies drop out of Netflix. And you know, it's like, Oh, that's going to disappear by the end of the month. Maybe that works for your home Netflix subscription. But that doesn't really work for a library collection.*" —P1

They also gave an illustrative example about academic journals.

> "*[A major publisher] doesn't promise you that they're always going to have the same journals, so you may sign a 3 year deal for 1,500 journals, but like every year some of those journals get sold or traded to other publishers. And so they're not part of the platform any more. And so … you're not 100% sure that you're gonna maintain access to it.*" —P1

As a result, libraries become dependent on those platforms:

> "*So we are completely dependent upon basically the benevolence of large multinational companies by and large, for developing web browsers that provide access to our web content. We have no say in how those app, those applications are built. We're dependent upon kind of an unsteady equilibrium between multiple companies.*" —P6

D.6 More on Theme 5 (Surveillance)

P5 elaborated some concrete examples of privacy risks and possible harms. Responding to a question about why library employees' access to long-lived patron records was concerning to them:

> "*Because anybody who's an employee or has who has access to that can see [those records] and can use [them] for all kinds of purposes, like, … if somebody is an abuser, for example, and is trying to find information about their child, or their partner, or whoever they're abusing, they can do that very easily, like see what books they've been checking out, what other items they've been checking out. A few years ago there was an instance of somebody who was a member of the Proud Boys … who was a library worker … who [looked up and] leaked a bunch of people's information …*" —P5

Government surveillance and censorship were another area of concern that they highlighted. Explaining that certain records in a library system that they were familiar with last indefinitely, such as books that were put on hold or returned late, P5 noted,

> "*something that I would be concerned [about], for example, would be like, oh, if there's a book that the police, for example, think that like that's a dangerous sign for [people] to read. . . . [I]f the police like wanted to see, like everybody who borrowed X book, they would have to have a subpoena. I believe they couldn't just come in and ask for that. . . . There's processes that they would have to follow technically [and] legally, but that . . . is in the realm of possibility.*" —P5

P3 noted that working for publicly funded groups can further raise legal concerns around data collection, intentionally or not, as *"most of the things we create are then public record"* and may be requested by the public.

Finally, data about reader preferences and history can be useful, both for libraries and for users. Four participants (P1, P5,P6,P9) mentioned both the potential utility of using such data and the privacy drawbacks of retaining it. From the patron perspective, P1 observed, "*it's sometimes pretty nice to log into a system and . . . remember the stuff that you read before*", and that "*some users love that*", "*but [that] also means now that vendor knows all about what you are doing . . . .*"

And from the libraries' perspective:

> "*[L]ibraries like to know how their books circulate, and they like to know a lot about the collections themselves. But for the most part they're able to do that without tracking a whole lot. I mean, knowing that a book circulated is a lot different than knowing who borrowed it, and how long and what they did with it. And so the basic approach across libraries for a very long time has been: we'll track when stuff comes in and when stuff goes out, but we're not going to keep track of anything more than what we need. . . . [L]ibraries have really buckled down on just saying we would prefer to not know or not retain records [due to privacy considerations].*" —P1

P1 described the above approach as heavily influenced by "*a pretty long history of*" libraries dealing with records requests from "*law enforcement or by other groups*" interested in "*finding out . . . What kind of books did they look at? And, you know, other things like that.*" All four participants here mentioned that collecting additional information would likely be "*useful, . . . for making collections decisions, for instance*" (P1), for "*educational and scholarly*" purposes (P6), and for promoting underutilized resources to certain patrons.

> "*There are, as in so many other cases, potential benefits in knowing what user behavior is like and providing enhanced services. . . . So you know, if I know what user populations, at least at some level of granularity, were consuming what, then I could better tailor my services to meet [their] needs . . . .*" —P9

At the same time, all four participants expressed serious privacy concerns about retaining much of that kind of data. While P5 notes that libraries they are familiar with *do* sometimes retain this kind of data, P1 "*think[s] libraries have been very hesitant about it, because of their [privacy] concerns.*"

> "*[T]he top priority is like, let's not screw up people's privacy along the way. And if we can learn interesting things that's great, particularly around what we should acquire in the future.*" —P1

D.7 Need for new technologies and approaches

Theme 6: Need for privacy in digital lending. P9 noted that their institution "*is trying to evaluate what future extension of digital lending activity it wants to undertake,*" and added:

> "*If there were . . . privacy protecting mechanisms that we are presently unaware of we would very much appreciate considering them in terms of architectural design. Because those elements have not often been represented . . .*" —P9

P6, responding to a question about processing of search queries locally or server-side, raised their own question, "*How do we ensure privacy from end to end?*" and suggested that existing solutions do not address this issue.

> "*[T]here is a gap right now in providing a secure methodology for access to books. That's not consistent across all of these different ebook providers. So that is, in fact, a real gap.*" —P6

P1 stated that innovations to improve privacy and transparency

> "*sound[] really cool, and I think [are] very, very needed. . . . the privacy concerns, I think, are among the bigger and more difficult ones to solve with the current licensing system.*" —P1

Theme 7: Need for accessibility via digital lending. Multiple participants mentioned that the library profession strives to provide better content accessibility to patrons. In P11's words:

> "*There's always a tension between our desire as library people and the reality. Right? So as library people, we want to provide more access to more people in more places.*" —P11

Six participants (P1, P4, P5, P6, P8, P11) discussed the accessibility benefits of offering digital access mechanisms in addition to physical ones. P6 emphasizes unmediated, efficient access to content:

> "*Our goal is to have a user walk up to their laptop or their phone and be able to search for a resource, find a resource, get the resource, and use the resource without a librarian or staff member intervening. So that's the goal. Let me try to do that for the vast majority without a human intervention.*" —P6

Not only could this be convenient for the patron, it may also reduce burden on libraries, as noted by P11.

> "*The process right now is pretty painful. There's a lot of manual steps. So we're thinking about how to automate those steps, how to make them easier.*" —P11
>
> "*And for the patron. Right? I mean, wouldn't it be nice if the patron didn't have to come into the reading room, if they could sit in the comfort of their own home and*

> *log on to something remote, and just get it that way. Like, why do we have to come into the [library] . . . ?*" —P11

P5 emphasized that different access mechanisms may serve different user needs, and that patrons have diverse preferences between, for example, print books, audiobooks, and e-books, as well as between talking to a librarian or looking up any of these resources online. P5 additionally noted that some patrons are "*pretty agnostic in terms of format*" and will go for whichever format provides access the soonest, highlighting another dimension in which offering diverse access mechanisms may increase accessibility.

> "*If they notice, oh, a book is going to take some time to get to me, whereas I can download the e-book or audiobook version right now, [they] would prefer right now, you know.*" —P5

Theme 8: Need for a paradigm shift and market diversity. Four participants (P1, P2, P6, P9) lamented the lack of available options other than digital licensing, especially for more resource-constrained libraries. P2 noted that "*[t]here are certain contexts in which licensing is appropriate, but licensing is not appropriate in every context*", explaining that licensing is not a substitute for other lending mechanisms but has incomparable properties,[19] and similarly that digital lending is not a substitute for physical lending. "*[Y]ou shouldn't be forced into only doing one thing*" (P11). P4 concurred with this sentiment, and they recalled the contrast between the original promise of digital lending and the current reality: "*[W]hen they were first introduced, it was: 'Oh my god, this is the future. There's not going to be any more print books.' ... I don't think print material's going to go away.*" P2 emphasized that the future of digital lending is

> "*not about getting better licenses. It's about encouraging a culture of ownership for institutions and choice in the market, so that everybody can have access to the same level of quality information.*" —P2

D.8 Interview Questions

We conducted semi-structured interviews based on the following questions, organized by topic area below.

Digital Lending and Digital Services.

(1) What digital services does your library provide?
   - Is there a system for lending digital books?
   - Is there a digital library catalog? Is the catalog only digital?
(2) Do you think it is a good idea to offer digital books for lending? Advantages and disadvantages?
(3) If your library offers lending of digital books:
   - Tell us about the system for lending digital books in your library. How familiar are you with how it works? How does it work? What do you like and dislike about it? Is there anything you feel like missing?
   - Do you remember how your library introduced the digital lending system? If so, what were the considerations when adopting a digital lending scheme, assuming there were different ones?
(4) If your library does not offer lending of digital books, whether and what are the specific reasons?
(5) What are the digital lending systems that you have heard of? What do you think of them?

Patron Information and Loan Records.

(1) When a patron joins a library for the first time, can patrons sign up online, or do they need to come to the library in person?
(2) Is there any identity check when patrons sign up? If so, what does the check consist of?
(3) When a patron joins a library, what patron information does the library collect and store?
   - Is all information required to join the library, or is some of it optional? Which pieces are required? Who has access to this information?
   - How is this information stored, e.g., on paper or using which digital platform?
   - Is this information used by the library for any purpose besides patron borrowing books?
(4) How do patrons use their account after registering?
(5) If a patron loses their library card, how do they get back into their account?
(6) What if a patron tries to register for multiple accounts? Whether and how could the system detect such behavior? Is this a problem you have encountered?
(7) About how many patrons does your library serve?
(8) About how many titles does your library offer for loan?
(9) About how many physical books does your library offer for loan?
(10) For about how many of your titles do you offer multiple copies for loan?
(11) If your library offers digital lending service:
   - How does the library decide which books to offer digitally?
   - About what fraction of the titles your library offers are available digitally?
   - About what fraction of the titles your library offers are available physically?
   - Are any titles offered only digitally or only physically?
   - Across the titles that are offered both physically and digitally, about what ratio would you estimate your catalog of digital to physical books is?
(12) About how many open loans does your library have?
(13) About how many books are loaned across a month-long period?
(14) How does a library keep records of open loans? For physical and/or digital books:
   - What does a record look like?
   - What information does the record contain?
   - Who has access to these records?
(15) Does the library keep records of closed loans from patrons?

[19] P2 offered examples of limited, specific situations where licensing could be a good fit: "*So I'm an academic library. I have a faculty member asking for one specific thing that will never be borrowed by any other faculty member ever again, likely that they only need for like one week of research, and then they never need again. . . . You shouldn't have to buy an entire book for that. . . . So that's one space in which licensing might be appropriate. . . . Or if you're a public library and you want to buy, you know, 400 copies of the new Colleen Hoover . . . [and] you know that after three months everyone's going to be reading the next Colleen Hoover. So you don't want to have to be paying for 400 copies of that forever. Maybe you want to be paying for 100. And so, you know, letting a license lapse, like buying a license upfront, and then letting [it] lapse and just having fewer copies, could be another reason why you would want to use licensing.*"

(16) If the library keeps records of closed loans:
- What a closed loan record would look like in the library system?
- Who can see these closed loan records?
- How long are these records kept?
- Why are these records kept? Does the library use these records?
- Are there any special circumstances where past loan records are deleted?

(17) If the library does not keep records of closed loans:
- How does a library delete the loan records?
- When does the deletion happen?
- Who can delete past loan records?
- Is it possible to "restore" deleted records under any circumstances?

(18) Do you have any idea or control over what (patron) data the publishers collect/mine/store about open and closed loans?

(19) Does the library use aggregated information of the titles? If so:
- What format of statistics?
- Who has access to it?
- How is this kind of statistics useful?

Library Policy Violations.

(1) What happens if a patron fails to return a book in time in the physical lending and/or digital lending?
(2) Other than not returning the book in time, are there other concerns about library policy violations by patrons?
(3) How often do disputes arise where a patron has a borrowed book according to the library records, but the patron claims they never borrowed that book or already returned it? When this happens, what is the process for resolving the dispute?
(4) If there are other violations, how does the library deal with it? Who is involved in it?

Publisher/Distributor Interactions.

(1) How does the library interact with rights holders of the books like publishers?
(2) Do you interact directly with publishers? If so, do they have any concerns about the so-called owned-to-loaned ratio?
(3) What are the concerns of publishers, if you know any?

E Additional Figures Related to Modeling

Fig. 1 summarizes three models of lending — traditional physical lending, digital licensing, and direct digital lending — in graphical form.

F Collision-Resistant Signatures

We require a non-standard property which we call *collision resistance* (CR), of our signature scheme—intuitively, that finding $\mathsf{vk}' \neq \mathsf{vk}$ such that signatures generated w.r.t. vk verify w.r.t. vk' is computationally hard. We define this property formally below.

Definition 2 (Collision-resistant signatures)**.** A signature scheme (KeyGen, Sign, Verify) is *collision resistant* if for any two-part PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, it holds that

$$\Pr\left[\begin{array}{ll} (\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) & \\ m \leftarrow \mathcal{A}_1(\mathsf{vk},\mathsf{sk}) & \\ \sigma \leftarrow \mathsf{SigSign}(\mathsf{sk},\mathrm{m}) & : \quad b = 1 \wedge \mathsf{vk} \neq \mathsf{vk}' \\ \mathsf{vk}' \leftarrow \mathcal{A}_2(\mathsf{vk},\mathsf{sk},m,\sigma) & \\ b \leftarrow \mathsf{SigVeri}(\mathsf{vk}',m,\sigma) & \end{array}\right] \leq \mathsf{ngl}(\lambda).$$

As noted in § 6, a CR signature scheme

$$\Sigma' = (\mathsf{KeyGen}', \mathsf{SigSign}', \mathsf{SigVeri}')$$

can be straightforwardly built from a standard existentially unforgeable signature scheme $\Sigma = (\mathsf{KeyGen}, \mathsf{SigSign}', \mathsf{SigVeri})$ and a collision-resistant hash function h (sampled from a CRHF) as follows:

- $\mathsf{KeyGen}'(1^\lambda)$ samples $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and outputs
$$(\mathsf{vk}',\mathsf{sk}') = (\mathsf{vk},(\mathsf{sk},\mathsf{vk}));$$
- $\mathsf{SigSign}'((\mathsf{sk},\mathsf{vk}),m)$ runs $\sigma \leftarrow \mathsf{SigSign}(\mathsf{sk},\mathrm{m})$ and outputs
$$\sigma' = (\sigma, h(\mathsf{vk})); \text{ and}$$
- $\mathsf{SigVeri}'(\mathsf{vk}',m,(\sigma,\eta))$ outputs $\top$ if
$$\eta = h(\mathsf{vk}) \wedge \mathsf{SigVeri}(\mathsf{vk}',m,\sigma)$$
and outputs $\bot$ otherwise.

We include a proof sketch showing that Σ' as defined above is indeed collision resistant.

Proof (sketch). Suppose, for contradiction, that we have an adversary $\mathcal{A}'$ that violates collision resistance of the signature scheme Σ' described above. We can then construct an adversary $\mathcal{A}_h$ that violates the collision resistance of the hash function h, as follows. $\mathcal{A}_h$ runs the experiment in Def. 2, obtaining outputs $(\mathsf{vk},\mathsf{sk})$, m, σ, and vk', then parses $\sigma = (\cdot,\eta)$. Notice that $\mathsf{SigVeri}(\mathsf{vk}',m,\sigma)$ by assumption on $\mathcal{A}$ — implying $h(\mathsf{vk}') = \eta$ by definition of $\mathsf{SigVeri}'$ — and $\mathsf{SigVeri}(\mathsf{vk},m\sigma)$ by correctness of the signature scheme — implying $h(\mathsf{vk}) = \eta$ by definition of $\mathsf{SigVeri}'$. Thus, we have a collision, violating collision resistance of h. □

The next proof sketch shows that Σ' as defined above is still existentially unforgeable provided Σ is existentially unforgeable.

Proof (sketch). Suppose, for contradiction, that we have an adversary $\mathcal{A}'$ that violates existential unforgeability of the signature scheme Σ' described above. We can then construct an adversary $\mathcal{A}$ that violates existential unforgeability of Σ, which takes as input an honestly generated verification key vk for Σ,

$$(\mathsf{vk},\cdot) \leftarrow \mathsf{SigSign}(\mathsf{sk},\mathrm{m}),$$

runs $(m',\sigma') \leftarrow \mathcal{A}'(\mathsf{vk})$, parses $\sigma' = (\sigma,\eta)$ and outputs (m',σ).

By construction of Σ', vk is a valid verification key for both Σ and Σ'. Let E, E' denote the events that $\mathsf{SigVeri}(\mathsf{vk},m',\sigma) = 1$ and $\mathsf{SigVeri}'(\mathsf{vk},m',\sigma') = 1$ respectively. $E' \Rightarrow E$, by construction of $\mathsf{SigVeri}'$. Moreover, by assumption on $\mathcal{A}'$, $\Pr[E']$ is non-negligible. Thus, $\mathcal{A}$ violates existential unforgeability of Σ. □

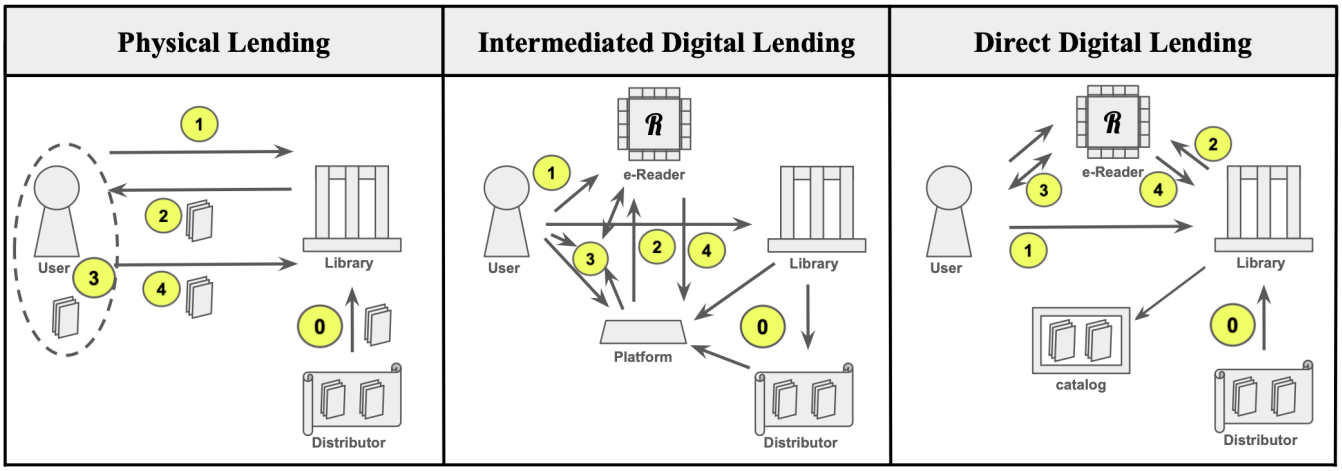

Figure 1: Three lending models: traditional physical book lending, licensing (in broad strokes), and direct digital lending. Above, (0) is purchasing of content, (1) is patron registration, (2) is lending, (3) is reading/accessing content, (4) is returning.

EdDSA. For our micro-benchmarks (§ 8) we use EdDSA signatures without applying the transformation Σ' (which does not add much overhead in the signature scheme itself, but would add noticeable overhead to our NIZK operations). EdDSA signatures consist of a hash (modelled in EdDSA's security analysis as a random oracle) applied to (a shift of) a base point of an elliptic curve, the public key, and the message. Unsurprisingly, this construction satisfies our notion of collision resistant signatures in the random oracle model. We give a brief sketch of the argument for collision resistance here.

Proof. Let (KeyGen, SigSign, SigVeri) be the EdDSA signature scheme. As a reminder, the EdDSA signature scheme works as so (with H as a random oracle):

- KeyGen generates a base point B on an elliptic curve and outputs as the secret key sk a random bitstring k and as the public key vk a point A on the curve determined by $H(k)$.
- SigSign generates a signature for message m by setting $R = H(H(k), m)B$ and $S = H(H(k), m) + H(R, A, m)B/A$. It then outputs (R, S).
- SigVeri computes for a given signature parsed as (R, S) and verification key A the hash $H(R, A, m)$ and ensures $8SB = 8R + 8H(R, A, m)A$.

Suppose toward contradiction there is an efficient strategy $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ such that $\mathcal{A}$ finds (m, vk') with noticeable probability.

When this occurs, we see that, for a given $vk = A$, $sk = k$, $\sigma = (R, S)$, the supplied $vk' = A'$, m must satisfy:

$$8SB = 8R + 8H(R, A', m)A'.$$

Noting that $SB = R + H(R, A, m)A$, we see this is equivalent to

$$H(R, A, m)A = H(R, A', m)A'.$$

For any choice of $A' \neq A$, though, the probability that H collides on these messages is negligible, contradicting the supposition. □

Why we need CR. We need CR signatures for our proof of *self-auditing soundness* (given in § J). Not all unforgeable signature schemes satisfy CR: to give a contrived toy example, in a signature scheme that ignores the last bit of the verification key, given (vk, sk), it is easy to find vk′ such that signatures w.r.t. vk also verify w.r.t. vk′, simply by flipping the last bit. It is not clear if there exist "natural" examples of secure signature schemes that are not collision resistant; nonetheless, we require CR as a definitional matter.

Though harmless in many applications of signatures, this is a real problem in our setting, where we seek to prevent users with distinct public keys from "sharing" loan records on the bulletin logBB, as this would enable overloaning. The self-audit procedure is, indeed, the one which checks individual loan records on e-readers against the public catalog logBB to detect overloaning.

G Complete Definitions

This section contains all of the definitions in our paper in one self-contained section for easy reference. As such, it repeats some definitions that were given in earlier sections.[20] (See also Table 2 for an index listing with threat modeling assumptions.)

Our highest level definition, of a *private, transparent digital lending system* is given in Def. 1. This definition modularly refers to three sub-definitions, Definitions 3 to 5. Each of Definitions 3 to 5, in turn, refer to further sub-definitions corresponding to properties such as correctness, soundness, and privacy for each functionality respectively, which are given in §§ G.1 to G.3.

Definition 1. Let LendLocked =
(patronregUL, readerregURL, borrowreqUL, readUR, addlogURL, trackreadR, returndelURL, downloadRL, auditlogV, verilogR)
be a tuple of efficient algorithms, let $\mathcal{L}$ be a party that implements their respective algorithms according to their syntax, and let $\mathcal{R}$ and $C(\cdot)$ be TEE machines which each implement their respective queries as according to Section 5. Then, we say that LendLocked is a *private, transparent digital lending system* if the appropriate algorithms of LendLocked:

- realize the register-borrow-return functionalities (Def. 3),

[20] Where a definition is repeated, we use the original definition number. As such, the numbering of definitions in this section is not an increasing sequence.

- realize the read functionality (Def. 4),
- realize the audit functionality (Def. 5),

and (readUR, trackreadR) satisfies copy deterrence (Def. 19).

Definition 3 (Register-borrow-return functionalities)**.** Let

$$(\mathsf{patronregUL}, \mathsf{readerregURL}, \mathsf{borrowreqUL}, \mathsf{readUR})$$

be a tuple of efficient algorithms. We say this tuple *realizes the register-borrow-return functionalities* if it satisfies Defs. 6 (borrowing correctness), 7 (borrowing soundness), 9 (returning correctness), 10 (returning soundness), and 11 (borrowing and returning privacy).

Definition 4 (Read functionality)**.** Let

$$(\mathsf{patronregUL}, \mathsf{borrowreqUL}, \mathsf{readUR}, \mathsf{trackreadR})$$

be a tuple of efficient algorithms. Then, we say this tuple *realizes the read functionality* if it satisfies Definitions 6 (reading correctness), 12 (reading soundness), and 13 (reading privacy).

Definition 5 (Audit functionality)**.** Let

$$(\mathsf{auditlogV}, \mathsf{verilogR}, \mathsf{addlogURL})$$

be a tuple of efficient algorithms. Then, we say this tuple *realizes the audit functionality* if it satisfies Definitions 14 (auditing correctness), 15 (auditing soundness), and 18 (auditing privacy).

G.1 Register-borrow-return functionalities

G.1.1 Borrowing and reading correctness. Our correctness definition (which only entails correctness of borrowing and reading, not of the public bulletin) is fairly straightforward from the intuitive functionality we want. That is, if:

(1) $\mathcal{U}$ is registered with $\mathcal{L}$,
(2) $\mathcal{U}$ has successfully borrowed a copy of b.title, and
(3) $\mathcal{U}$ queries readUR on a given index while the current time t is less than the return deadline d,

then $\mathcal{U}$ gets back the respective piece of text of the book.

Note that this abstracts out various other parameters that might affect the ability to borrow a book (e.g., limits on how many books can be borrowed at once, age restrictions, whether the library has b.title in stock). These are all non-security parameters that can be checked within a call of borrowreqUL, and so we consider these requirements as out of scope.

Definition 6 (Borrowing and reading correctness)**.** We say that (patronregUL, borrowreqUL, readUR) satisfies *borrowing and reading correctness* if, for any copy with publication identifier b.title, we have:

$$\Pr\left[\mathsf{b.cont}[i] = \tau \;\middle|\; \begin{array}{r} \mathsf{patronregUL}(\mathsf{uid}) = \mathsf{k}_u, \\ \mathsf{borrowreqUL}(\mathsf{uid}, \mathsf{k}_u, \mathsf{b.title}) = d \\ C(\cdot) = t, t \le d, \\ \mathsf{readUR}(\mathsf{uid}, \mathsf{k}_u, \mathsf{b.title}, i, t) = \tau \end{array}\right] = 1. \quad (1)$$

G.1.2 Borrowing soundness. We will define soundness briefly, noting that in this setting we are considering a malicious user $\mathcal{U}$, and every other party honest. Note that in our constructions, these are all ensured at the protocol level.

Definition 7 (Borrowing soundness)**.** Let $\lambda \in \mathbb{N}$. Consider any set of honest registered users $\mathfrak{U}$. We say that (patronregUL, borrowreqUL) satisfies *borrowing soundness* if, for any PPT $\tilde{\mathcal{U}}$ who has not registered with the library and efficiently generates a tuple of borrowing credentials $(\tilde{\mathsf{k}_u}, \tilde{k_{\mathcal{U},\mathcal{R}}}, \mathsf{b.title})$, $\tilde{\mathcal{U}}$ cannot borrow any book. That is,

$$\Pr\left[\mathsf{borrowreqUL}(\mathsf{uid}, \tilde{\mathsf{k}_u}, \tilde{k_{\mathcal{U},\mathcal{R}}}, \mathsf{b.title}) \neq \perp\right] \le \mathsf{ngl}(\lambda), \quad (2)$$

where the randomness is over the choices of $\tilde{\mathcal{U}}$.

G.1.3 Returning Correctness. In order to understand what it means for a return to be "correct," we must introduce the concept of an *open loan* to denote when a user should and shouldn't have access to a piece of content. At a high level, an open loan of a book b.title represents the user having access to that book. A user has an open loan for a book if and only if it has been borrowed and not returned (either from triggering a return manually or from the deadline for the borrow passing).

Definition 8 (Open loan)**.** Let b.title be an identifier for a book, and let logBB be a properly formed bulletin. For any given user $\mathcal{U}$ with id uid and public user-reader key $\mathsf{vk}_{\mathsf{ur}}$, we say that $\mathcal{U}$ has an *open loan* for b.title, denoted by

$$\mathsf{openloan}(\mathsf{logBB}, \mathsf{uid}, \mathsf{b.title}) = 1,$$

if logBB contains a valid "borrow log entry" $\ell \leftarrow$ borrowreqUL from $\mathcal{U}$ for b.title but no corresponding "return log entry" $\ell' \leftarrow$ returndelURL from $\mathcal{U}$ for b.title, where ℓ, ℓ' denote the log entry/ies that borrowreqUL, returndelURL append to logBB respectively. Otherwise, we say that $\mathcal{U}$ does not have an open loan for b.title, that is, $\mathsf{openloan}(\mathsf{logBB}, \mathsf{uid}, \mathsf{b.title}) = 0$.

Note that openloan is not efficiently computable given logBB, uid, and b.title; indeed, this is a privacy property we enforce by design. The concept of an open loan is a useful stepping stone in our proofs; it does not need to be efficiently computable. We use it only as an auxiliary definition for reasoning about returning correctness and reading soundness.

Returning correctness requires that when a return is triggered, the book is actually returned, closing an open loan.

Definition 9 (Returning correctness)**.** We say returndelURL satisfies *returning correctness* if, for any honest user $\mathcal{U}$ with id uid and keys k_u, k_{ur} with honest reader $\mathcal{R}$, for any valid bulletin state logBB such that $\mathsf{openloan}(\mathsf{logBB}, \mathcal{U}, \mathsf{b.title}) = 1$, it holds that

$$\mathsf{openloan}(\mathsf{logBB}', \mathcal{U}, \mathsf{b.title}) = 0$$

where logBB′ denotes the bulletin state after the procedure

$$\mathsf{returndelURL}(\mathsf{uid}, \mathsf{k}_u, \mathsf{k}_{ur}, \mathsf{b.title})$$

is run starting at bulletin state logBB.

G.1.4 Returning soundness. Returning soundness requires that any book that has been returned cannot be read.

Definition 10 (Returning soundness)**.** Let $\lambda \in \mathbb{N}$. We say that readUR satisfies *returning soundness* if, for any malicious PPT algorithm $\mathcal{U}$ with id uid and keys k_u, k_{ur}, any book b.title, index i, and

bulletin logBB,

$$\Pr\left[\mathsf{b.cont}[i] = \tau \middle| \mathsf{readUR}(\mathsf{logBB}', \mathsf{uid}, \mathsf{k}_u, \mathsf{b.title}, i, t) = \tau\right] \leq \mathsf{ngl}(\lambda) \tag{3}$$

where t is any timestamp as chosen by $\mathcal{U}$ and logBB' is the bulletin state after

$$\mathsf{returndelURL}(\mathsf{uid}, \mathsf{k}_u, \mathsf{k}_{ur}, \mathsf{b.title}) \tag{4}$$

is run starting at bulletin state logBB.

G.1.5 Borrowing and returning privacy.

Definition 11 (Borrowing and returning privacy)**.** Let $\lambda \in \mathbb{N}$. We say that (borrowreqUL, returndelURL) satisfy *borrowing and returning privacy* if, for any uid, k_u, for any borrow request

$$\rho = \mathsf{borrowreqUL}(\mathsf{uid}, \mathsf{k}_u, k_{\mathcal{U},\mathcal{R}}, \mathsf{b.title})$$

at time ts and log state logBB_0 prior to this borrow request, and for any corresponding return request

$$\rho' = \mathsf{returndelURL}(\mathsf{uid}, \mathsf{k}_u, k_{\mathcal{U},\mathcal{R}}, \mathsf{b.title})$$

at time $ts' > ts$, with a (possibly empty) series of intermediate log entries logBB_1 after the borrow request and before the return request, for any PPT adversary $\mathcal{A}$ with access to logBB (after the borrow and return requests), there exists a simulator $\mathcal{S}$ which, given only b.title, logBB_0, logBB_1, ts and ts', can produce a simulated borrow and return request χ, χ' that are indistinguishable to $\mathcal{A}$ given the full catalog context. That is, the following two distributions are computationally indistinguishable:

$$\mathcal{A}(\mathsf{logBB}_0||\rho||\mathsf{logBB}_1||\rho') \text{ and } \mathcal{A}(\mathsf{logBB}_0||\chi||\mathsf{logBB}_1||\chi'),$$

where $(\chi, \chi') \leftarrow \mathcal{S}(\mathsf{b.title}, \mathsf{logBB}_0, \mathsf{logBB}_1, ts, ts')$.

G.2 Read functionality

Correctness of the reading protocol is already defined above, together with borrowing correctness (Def. 6 in § G.1).

G.2.1 Reading soundness. Reading soundness requires that users cannot load the content of books they have not borrowed. As a look ahead, our system will satisfy this using trusted hardware to enforce the reader to enforce the check on open loans.

Definition 12 (Reading soundness)**.** Let $\lambda \in \mathbb{N}$. We say that readUR satisfies *reading soundness* if, for any malicious PPT algorithm $\mathcal{U}$ and honest reader $\mathcal{R}$ and any book b.title, if $\mathcal{U}$ does not have an open loan for b.title (as per Definition 8), then

$$\Pr\left[\mathsf{b.cont}[i] = \tau \middle| \mathsf{readUR}(\mathsf{uid}, \mathsf{k}_u, \mathsf{b.title}, i, t) = \tau\right] \leq \mathsf{ngl}(\lambda) \tag{5}$$

G.2.2 Reading privacy. In the physical world, privacy over how someone reads a book is taken for granted as obvious. However, as we see in licensing and other digital lending schemes, this is not necessarily obvious. Namely, in an online scheme, where the user must constantly or occasionally be in contact with the library or a third party, there may be leakage to this third party over the identity of who is accessing what book. Note that in our scheme this is obvious by the fact that the user and reader only communicate with a physical channel (since the reader is an object the user interacts with), and they only interact with each other for readUR.

Definition 13 (Reading privacy)**.** Let $n \in \mathbb{N}$. For any PPT adversary $\mathcal{A}$, for any sequence of $i_1, \ldots, i_n$, we have that no party other than $\mathcal{U}$ and $\mathcal{R}$ can tell when a book has been accessed or not. That is:

$$|\Pr[b \leftarrow \mathcal{A}(\mathsf{k}_u, \mathsf{k}_r, \mathsf{logBB}, \mathsf{b.title})|\mathsf{readUR}(\mathsf{uid}, \mathsf{k}_u, \mathsf{b.title}, i_1), \ldots]$$

$$-\Pr[b \leftarrow \mathcal{A}(\mathsf{k}_u, \mathsf{k}_r, \mathsf{logBB}, \mathsf{b.title})|\emptyset]| \leq \mathsf{ngl}(\lambda),$$

where $\emptyset$ denotes an empty set of access requests (i.e., no accesses have occurred), and '...' refers to the queries

$$\mathsf{readUR}(\mathsf{uid}, \mathsf{k}_u, \mathsf{b.title}, i_1), \ldots, \mathsf{readUR}(\mathsf{uid}, \mathsf{k}_u, \mathsf{b.title}, i_n)$$

made in sequence.

G.3 Audit functionality

G.3.1 Auditing correctness. Our auditing correctness definition encompasses correctness for both public-auditing (auditlogV) and self-auditing (verilogR).

Definition 14 (Auditing correctness)**.** Let $k, \lambda \in \mathbb{N}$, and let

$$\{(\mathsf{ts}_i, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)\}_{i\in[k]}$$

be a sequence of k tuples containing honestly generated user-reader keys $\mathsf{k}_{ur,i}$ and honestly generated signatures σ_i that complies with the overloaning policy DistLst: i.e., for all b.title, we have

$$|\{i \mid i \in [k], \mathsf{b.title}_i = \mathsf{b.title}\}| \leq \mathsf{DistLst}[\mathsf{b.title}].$$

Let $\mathfrak{C} = \{\mathsf{k}_{ur,i}\}$ be the set of unique user-reader credentials in this sequence, and let $\mathfrak{R}$ be a set of $|\mathfrak{C}|$ e-readers each containing distinct credentials in $\mathfrak{C}$, Let logBB be the bulletin formed by appending a series of borrow and return log entries under this series of credentials, generated honestly by the protocol

$$\mathsf{addlogURL}(\mathsf{ts}_i, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i),$$

including interaction with the corresponding e-readers in $\mathfrak{R}$ for each credential set $\mathsf{k}_{ur,i}$. Moreover, let each return request in this sequence correspond to a unique previously borrowed book under the same credentials. Then, we say that logBB satisfies *auditing correctness* if it satisfies *public-auditing correctness* (6) and *self-auditing correctness* (7).

$$\Pr[\mathsf{auditlogV}(\mathsf{logBB}) = 1] = 1 \tag{6}$$

$$\Pr[\mathsf{verilogR}(\mathsf{ts}, \mathsf{uid}, \mathsf{b.title}, \mathsf{k}_{ur}, \sigma, \mathsf{logBB}) = \top] = 1. \tag{7}$$

G.3.2 Auditing soundness. Our notion of auditing soundness is, at a high level, meant to prevent a malicious library and user from misrepresenting the library's loans assuming honest e-reader behavior. We model this by having a public append-only log (in our syntax, this is logBB) which the library writes each borrow or return query to, with each being signed by the reader. Anyone with view of logBB should be able to verify that the log attests to the library claiming it is only loaning as many pieces of content as it is authorized to at any one time — and we can guarantee this by showing log entries are collision-resistant (Definition 17). Then, each reader should be able to verify its own (and only its own) entries on the log (Definition 16). Combining these properties with this workflow, then, we can ensure that (a) the log accurately reflects the state of the library's open and closed loans, and (b) each entry on the log accurately reflects one and only one reader's open loan. Compiling these definitions, then, we get our notion of auditing soundness (Definition 15, below).

Definition 15 (Auditing soundness)**.** Let logBB be an append-only log with entries appended through some process addlogURL. Then, we say logBB satisfies *auditing soundness* if it satisfies self-auditing soundness (Definition 16) and log collision-resistance (Definition 17).

We can now introduce the constituent definitions.

Definition 16 (Self-auditing soundness)**.** Let $k, \lambda \in \mathbb{N}$, and let $\{(\mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)\}_{i\in[k]}$ be a sequence of k credentials, and let logBB be the bulletin formed by appending the output of

$$\mathsf{addlogURL}(\mathsf{ts}, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)$$

in sequence for $i \in [k]$. Let $\mathsf{k}_{ur,i} = (\mathsf{epk}_{,i}, \cdot, \mathsf{vk}_{\mathsf{ur},i}, \cdot)$. Then, we say that logBB satisfies *self-auditing soundness* if, for any PPT algorithm outputting a tuple of the form

$$\begin{aligned}&(\mathsf{ts}', \mathsf{uid}', \mathsf{b.title}', (\mathsf{epk}', \mathsf{vk}'_{\mathsf{ur}}))\\ \notin &\left\{(\mathsf{ts}_{,i}, \mathsf{uid}_i, \mathsf{b.title}_i, (\mathsf{epk}_{,i}, \mathsf{vk}_{\mathsf{ur},i}))\right\}_{i\in[k]}\end{aligned} \tag{8}$$

not in the sequence of credentials in logBB, the audit will fail with overwhelming probability. That is:

$$\Pr\left[\mathsf{verilogR}(\mathsf{ts}', \mathsf{uid}', \mathsf{b.title}', \mathsf{k}'_{ur}, \mathsf{logBB}) = \top\right] \leq \mathsf{ngl}(\lambda). \tag{9}$$

Definition 17 (Log collision-resistance)**.** Let $\lambda \in \mathbb{N}$, and let KeyGen be the user key distribution over $\{0,1\}^\lambda$. We say that logBB is *collision-resistant* if, for all $(\mathsf{uid}, \mathsf{b.title}) \neq (\mathsf{uid}', \mathsf{b.title}')$, we have:

$$\begin{aligned}\Pr\big[&\mathsf{addlogURL}(\mathsf{ts}, \mathsf{uid}, \mathsf{b.title}, \mathsf{k}_{ur}, \sigma)\\ &= \mathsf{addlogURL}(\mathsf{ts}', \mathsf{uid}', \mathsf{b.title}', \mathsf{k}'_{ur}, \sigma')\big]\end{aligned}$$

is negligible, where $\mathsf{k}_{ur}, \mathsf{k}'_{ur}$ are generated uniformly from KeyGen and σ, σ' are honestly generated signatures on the messages using keys $\mathsf{k}_{ur}, \mathsf{k}'_{ur}$ respectively, according to the protocol description.

G.3.3 Auditing privacy. Our notion of auditing privacy ensures that no adversary can gain more information from the publicly accessible logBB than the titles and timestamps of books borrowed or returned. Note that this matches the physical setting, as a motivated adversary who checks the library's catalogs repeatedly would be able to learn this same information (but not, for example, the identities of who is borrowing or returning books).

Definition 18 (Auditing privacy)**.** Let $k, \lambda \in \mathbb{N}$, and let

$$\{(\mathsf{ts}_i, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)\}_{i\in[k]}$$

be a sequence of k credentials, and let logBB be the bulletin formed by appending $\ell = \mathsf{addlogURL}(\mathsf{ts}_i, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)$ in sequence. Let $info = \{\mathsf{b.title}_i, \mathsf{ts}_i\}_{i\in[k]}$ be the sequence's books borrowed/returned and timestamps of each record. We say that our scheme satisfies *auditing privacy* if, for all PPT adversaries $\mathcal{A}$, there exists a simulator Sim such that $REAL(\lambda) \approx SIM(\lambda)$, where

$$REAL(\lambda) = \{out \mid out \leftarrow \mathcal{A}(\mathsf{logBB})\} \text{ and}$$

$$SIM(\lambda) = \{out \mid out \leftarrow \mathcal{A}(\mathsf{logBB}), \mathsf{logBB} \leftarrow \mathsf{Sim}(info)\}.$$

G.4 Copy deterrence

Recall that copy deterrence is an auxiliary definition that we formulate to try to model the practical difficulty of creating copies of books in the physical setting — a moderate difficulty notion, as copying *is* possible, but onerous. The moderate nature of the definition may limit the practical utility of the copy deterrence against determined adversaries; however, we believe the definition is nonetheless interesting and non-trivial. In particular, it aims to capture a notion as similar as possible to the physical setting, where copying is similarly possible, but onerous, and impossible to perfectly prevent. Moreover, similarly to the physical setting, while copy deterrence may not be effective against a single adversary determined to make a single copy, it creates a greater practical barrier to copying at scale.

We found formally modeling copy deterrence to be quite interesting and nuanced. Because the goal of borrowing content is in theory to read it, and because one could simply write down every word they read as they read it, we cannot – and could never, digitally or physically – entirely prevent a motivated adversary from copying a book, potentially without anyone else knowing, based on just performing allowed queries that are identical to those needed to simply read the book. Note that practical methods of surveilling the user do not mitigate this problem.[21]

Moreover, an adversary may have some side information (say, well-known quotes, the last few pages of a book, an LLM trained on the book, or even just knowledge of story conventions) which they can leverage to make a guess at any small portion they do not read. Further, a given author's voice and word choice may give clues to the distribution of other text by the same author, so even without such "side information," someone reading a book may be able to gain information on pieces of the text they have not yet read. Unread portions will, therefore, never realistically be truly "unpredictable" in a strong sense that is similar to usual cryptographic definitions; by their nature, books just aren't unpredictable like that.

These subtleties make our formalization of copy deterrence rather delicate. We consider a distinguisher who wants to understand whether they are reading an e-book in our scheme (the $\mathcal{R}e$ queries below) or reading an idealized e-book (the $\mathcal{I}d$ queries below). By idealized, we mean that the access to the book is essentially as if it was a physical book. That is, you can only read the currently open pages, and trying to divine anything else (without turning the pages) gives nothing useful. This neatly allows for "side information" and predictability without modeling them explicitly.

Definition 19 (Copy deterrence)**.** Let $\lambda \in \mathbb{N}$, and let D be PPT. Let $\mathsf{uid}, \mathsf{k}_u$ be arbitrary. Let Sim be PPT. We define the following real and ideal games, denoted $\mathcal{R}e$ and $\mathcal{I}d$.

- $\mathcal{R}e$ maintains an instance of $\mathcal{R}$ loaded with b.cont of b.title belonging to uid with key k_u. It takes two queries:
  (1) On query $\mathsf{readUR}^{\mathcal{R}e}(i)$, $\mathcal{R}e$ queries trackreadR to $\mathcal{R}$ (updating the readers state) and then queries $\mathsf{readUR}(\mathsf{uid}, \mathsf{k}_u, i)$ and returns its response.
  (2) On query $\mathsf{dump}^{\mathcal{R}e}(\cdot)$, $\mathcal{R}e$ returns $\mathcal{R}_{pub}$, the contents of the public memory of $\mathcal{R}$.

[21] Having someone constantly watching and following the patron everywhere might make a difference, but that is outside our scope as it is wildly impractical and unacceptably intrusive. Also, traditional library lending never employed such surveillance.

- $\mathcal{I}d$ maintains Sim and initializes it with uid, b.title, k_u. It the following queries:
  (1) On query $\text{readUR}^{\mathcal{I}d}(i)$, $\mathcal{I}d$ queries $\mathcal{F}$, the ideal functionality, which returns solely b.cont$[i]$ for the stored book b.title. It returns this and also forwards the response to Sim.
  (2) On query $\text{dump}^{\mathcal{I}d}(\cdot)$, $\mathcal{I}d$ queries Sim, which must respond.

We say that readUR satisfies *copy deterrence* if there exists a PPT simulator Sim such that, for all copies with title b.title and content b.cont, and all PPT D, we have

$$\left| \begin{array}{l} \Pr\left[D^{\text{readUR}^{\mathcal{R}e},\text{dump}^{\mathcal{R}e}}(\text{uid}, k_u, \text{b.title}) = 1\right] \\ \\ -\Pr\left[D^{\text{readUR}^{\mathcal{I}d},\text{dump}^{\mathcal{I}d}}(\text{uid}, k_u, \text{b.title}) = 1\right] \end{array} \right| \leq \text{ngl}(\lambda),$$

where probabilities are over $\mathcal{R}$'s randomness and Sim, respectively.

H Definitions of Hiding, Binding, and Collision Resistance

We refer to Katz & Lindell's classic textbook [35], particularly Chapter 5 about hash functions (syntax defined below) and commitment schemes (syntax defined in §6). We give standard definitions of collision resistance (for hash functions) and hiding and binding (for commitment schemes) below.

Definition 20. A *hash function* is a pair of probabilistic polynomial-time (PPT) algorithms $\Pi = (\text{Gen}, H)$ such that:

- Gen is a probabilistic algorithm which on input 1^λ outputs a key s.
- H is a deterministic algorithm that takes as input a key s and a string $x \in \{0,1\}^*$, and outputs $H^s(x) \in \{0,1\}^{\ell(\lambda)}$, where ℓ is the output length (for fixed-length variants, input is restricted to $\{0,1\}^{\ell'(\lambda)}$ with $\ell'(\lambda) > \ell(\lambda)$).

Definition 21 (Collision Resistance)**.** Let the *collision-finding experiment* $\text{Collision}_{\mathcal{A},\Pi}(n)$ be defined as follows.

(1) A key s is generated: $s \leftarrow \text{Gen}(1^\lambda)$.
(2) The adversary $\mathcal{A}$ is given s and outputs x, x'.
(3) The output of the experiment is 1 if and only if $x \neq x'$ and $H^s(x) = H^s(x')$.

A hash function $\Pi = (\text{Gen}, H)$ is *collision resistant* if for every PPT adversary $\mathcal{A}$ there exists a negligible function ngl such that

$$\Pr\left[\text{Collision}_{\mathcal{A},\Pi}(n) = 1\right] \leq \text{ngl}.$$

Definition 22 (Hiding)**.** Let $\Pi = (\text{Gen}, \text{Com})$ be a commitment scheme. Let the *hiding experiment* $\text{Hiding}_{\mathcal{A},\Pi}(n)$ be defined as follows.

(1) Run $pp \leftarrow \text{Gen}(1^\lambda)$ and give pp to $\mathcal{A}$.
(2) $\mathcal{A}$ outputs a pair of messages m_0, m_1 of equal length.
(3) A uniform bit $b \in \{0,1\}$ is chosen, and randomness r is chosen uniformly. Compute
$$c := \text{Com}(pp, m_b; r)$$
and give c to $\mathcal{A}$.
(4) $\mathcal{A}$ outputs a bit b'. The experiment outputs 1 if and only if $b' = b$.

Algorithm 1 Register Protocol

```
# Library registers new patrons
procedure PatronRegL(u.info, uauth)
    uid ← $ and uid ∉ st_L.ulst          ▷ Generate unique user identifier
    st_L.ulst ← st_L.ulst + (u.info, uid, uauth) return uid, st_L.ulst
# User initiates registration request
procedure GenRegReqU(uid, st_U.auth)
    st_U.auth ← st_U.auth + uid
    regmsg = (u.s, uid, uauth) return regmsg, st_U.auth
# Reader generates key pairs for registration
procedure KeyGenRegR(1^λ, regmsg, st_R.key)
    if                                   ▷ No double registration
        st_R.key  then return ERROR already registered
    (vk_ur, sk_ur) ← KDF(1^λ, u.s||st_R.s)
    regreq ← (uid, uauth, vk_ur)
    return regreq, st_R.key ← (vk_ur, sk_ur)
# Library processes registration request
procedure ProcessRegReqL(regreq, st_L.reglst, logBB)
    (uid, uauth, vk_ur) ← regreq

    if          ▷ Check patron membership, commit and append vk_ur
        (uid, uauth) ∈ st_L.ulst  then
        com ← Commit(vk_ur), com'_vk ← Append(vk_ur, com_vk)
        logBB ← logBB + (vk_ur, com'_vk)
        regres ← vkref where logBB[vkref] == (vk_ur, com'_vk)
    return regres, st_L.reglst, logBB
# Reader verifies its key pair correctly appended
procedure FinishRegR(regres, st_R.key)
    vkref ← regres
    if          ▷ Verify vk_ur correctly committed and appended to logBB
        AppendVeri(logBB [vkref ].com, logBB [vkref ].vk_ur) == ⊤ re-
        turn ⊤  then
      return ⊥, st_R.key ← st_R.key − (vk_ur, sk_ur)
```

Π is *hiding* if there exists a negligible function ngl such that for every adversary $\mathcal{A}$ and every n,

$$\Pr\left[\text{Hiding}_{\mathcal{A},\Pi}(n) = 1\right] \leq \tfrac{1}{2} + \text{ngl}.$$

Definition 23 (Binding)**.** Let the

binding experiment $\text{Binding}_{\mathcal{A},\Pi}(n)$. be defined as follows.

(1) Run $s \leftarrow \text{Gen}(1^n)$ and give s to $\mathcal{A}$.
(2) $\mathcal{A}$ outputs (c, m, r, m', r').
(3) The experiment outputs 1 iff $m \neq m'$ and $\text{Com}_s(m; r) = c = \text{Com}_s(m'; r')$.

Π is *binding* if for every PPT adversary $\mathcal{A}$ there exists a negligible function ngl such that

$$\Pr\left[\text{Binding}_{\mathcal{A},\Pi}(n) = 1\right] \leq \text{ngl}.$$

I Additional Algorithm Specifications

I.1 List of State Variables

Table 5 summarizes the internal state for $\mathcal{U}$, $\mathcal{R}$, and $\mathcal{L}$.

I.2 Register Protocol

The detailed register protocol is in Algorithm 1.

Table 5: Summary of State Variables.

State	Included Items	Notation	Explanation
st_U	user authentication info	st_U.auth	(u.info, u.s, uauth, uid)
	ephemeral info per loan	st_U.eph	(b.title, ts, epk, esk, σ_{eph})
st_R	Stored load request for later verification	st_R.req	(b.title, ts, epk, σ_{eph}, π_{loan}, uauth)
	$\mathcal{R}$ built-in secret	st_R.s	Unique per $\mathcal{R}$
	$\mathcal{U}$-$\mathcal{R}$ key pair	st_R.key	A key pair (vk_{ur}, sk_{ur})
	local database for downloaded content	st_R.ctdb	(b.title, logref, ddl, b.cont)
st_L	list of registered $\mathcal{U}$-$\mathcal{R}$ verification keys	st_L.reglst	(uid, vk_{ur})
	list of authorized patron as users	st_L.ulst	(uid, uauth)
	loan policy of $\mathcal{L}$	st_L.rul	E.g., an upper bound number of open loans per user
	open loan database	st_L.loan	(b.title, uid, ddl, $\top/\bot$)

I.3 Borrow Protocol

The detailed register protocol is in Algorithm 2.

I.4 Access Protocol

The detailed access protocol is in Algorithm 3.

I.5 Return Protocol

The detailed return protocol is in Algorithm 4.

I.6 Audit Protocol

The detailed register protocol is in Algorithm 5.

J Remaining Proofs

The purpose of this section is to prove our main theorem, which was also stated in Section 7), and is restated below.

Theorem 1. *LendLocked (Section 6) is a private, transparent digital lending system (Definition 1), when it is instantiated with a secure KDF, signature scheme, NIZK, commitment scheme, and collision-resistant hash function, and with a secure TEE $\mathcal{R}$.*[22]

The rest of this section will prove this theorem, splitting the proof modularly as Definition 1 does. We recall our three tiers of syntax from Section 5 and Table 1. The proofs in this section demonstrate how the LendLocked protocol (defined using the third tier of syntax in Table 1), when mapped to our definitional syntax (that is, the second tier in Table 1), satisfies our definitions (from Section 5.2). This yields our desired high-level functionalities (expressed in the first tier of syntax in Table 1).

As this section frequently references definitions given earlier, we remind the reader that a complete listing of definitions is available in § G.

J.1 Proving Register-Borrow-Return Functionalities

Lemma 2. *The constructions of patronregUL, readerregURL, borrowreqUL, and readUR in Section 6 realize the register-borrow-return functionalities. That is, they satisfy Definition 3.*

[22]The CRHF is needed for the Merkle commitment, not used directly in the construction. See Section 6 for definitions of these cryptographic primitives and the security properties that LendLocked requires from each. See Section 5 for our TEE assumption.

Proof. We must prove correctness of borrowing and returning (Def. 6 and Def. 9), soundness of borrowing and returning (Definitions 7 and 10), and privacy of borrowing and returning (Def. 11) addressed in turn below.

Borrowing and Reading Correctness (Def. 6). By the protocol definition, if borrowreqUL returns d (Eq. (1), lines 1–2), then $\mathcal{U}$ has an open loan on b.title until deadline d, and $\mathcal{L}$ will load the deadline and initial book contents into the e-reader. After this, while $t \le d$, when the user requests readUR from the e-reader (Eq. (1), lines 3–4), the e-reader checks that the current time $t \le d$ (Algorithm 3, Algorithm 3) and that their user still has an open loan on b.title until deadline d according to logBB (Algorithm 3, Algorithm 3); if both conditions are true, then the e-reader either loads pre-downloaded book content if available or generates a request for additional content to be sent to the library (Algorithm 3, Algorithm 3). In the former (pre-downloaded) case, Eq. (1)'s requirement that the e-reader correctly loads the requested book content is satisfied, as this definition assumes book content provided by the library is accurate. In the latter (library request) case, it remains to show how the library responds to this request: it too verifies that the currently time $t \le d$ (Algorithm 3, Algorithm 3) that the user making the request is a valid registered user, and that that user has an open loan on the requested material according to the library's internal records (Algorithm 3, Algorithm 3), and if all these conditions are satisfied, returns the requested book content (Algorithm 3, Algorithm 3). As Def. 6 assumes the user/reader behave honestly and Eq. (1) is conditioned on $t \le d$ (Eq. (1), line 3), we have that all the conditions above hold, and so that the e-reader correctly loads the requested book content as required.

Borrowing Soundness (Def. 7). Let $\lambda \in \mathbb{N}$. Consider any set of honest registered users $\mathfrak{U}$. As part of LendLocked's registration process, the honestly generated public key of each valid user-reader pair is committed to using a Merkle-Tree-based commitment scheme (Algorithm 1, Algorithm 1). Let $\tilde{\mathcal{U}}$ be a PPT entity that has not registered with the library, and let

$$(\tilde{k}_u, \tilde{k}_{\mathcal{U},\mathcal{R}}, \text{b.title}) \leftarrow \tilde{\mathcal{U}},$$

as in Def. 7. Furthermore, let $\tilde{k}_u := (\tilde{k}^{pub}_{\mathcal{U},\mathcal{R}}, \tilde{k}^{priv}_{\mathcal{U},\mathcal{R}})$, i.e., consider the public (verification) and private (signing) keys separately.

Algorithm 2 Borrow Protocol

User initiates loan request
procedure GenLoanMsgU(1^λ, b.title, st_U.auth)
  (epk, esk) ← KeyGen(1^λ) ▷ *Generate ephemeral key pair*
  ts ← $C(\cdot)$ ▷ *Get current time*
  σ_{eph} ← Sign(esk, b.title || ts) ▷ *Sign a fresh loan message*
  loanmsg ← (b.title, ts, epk, σ_{eph}, uid, uauth)
  st_U.eph ← st_U.eph + (loanmsg, esk) ▷ *Store ephemeral values*
  return loanmsg, st_U.eph
Reader generates loan-specific proof for this request
procedure GenLoanReqR(loanmsg, st_R.key)
  (b.title, ts, epk, σ_{eph}, uid, uauth) ← loanmsg ▷ *Parse loanmsg*
  if ▷ *Ensure valid σ_{eph}*
    ⊥ ← SigVeri(epk, σ_{eph}, b.title ||ts) **then return** ERROR invalid signature σ_{eph}
  (vk_{ur}, sk_{ur}) ← st_R.key ▷ *Fetch $\mathcal{U}$-$\mathcal{R}$ key pair from st_R*
  π_{loan} ← NIZK{(vk_{ur}, sk_{ur}, σ_{ur}, π_{mem}, epk, σ_{eph}) :
    σ_{ur} = SigSign(sk_{ur}, b.title ||ts ||epk)∧
    ⊤ = SigVeri(vk_{ur}, σ_{ur}, b.title ||ts ||epk)∧
    ⊤ = MemberVeri(vk_{ur}, π_{mem}, logBB.com)∧
    ⊤ = SigVeri(epk, b.title ||ts, σ_{eph})}, ▷ *Generate NIZK for loan on logBB*
  h ← $H(\sigma_{ur}, \sigma_{eph})$
  loanreq ← (b.title, ts, epk, σ_{eph}, π_{loan}, uid, uauth, h)
  st_R.req ← loanreq ▷ *Store loanreq for later VerLoanResR()*
  return loanreq, st_R.req
Library verifies and appends loan request upon approval
procedure ProcessLoanReqL(loanreq, logBB, st_L.rul)
  (b.title, ts, epk, σ_{eph}, π_{loan}, uid, uauth, h) ← loanreq ▷ *Parse loanreq*
  if
    (uid, uauth) ∈ st_L.ulst ▷ *Verify $\mathcal{U}$ authorization*
    |st_L.loan[b.title] + 1| ≤ DistLst[b.title] ▷ *b.title is available*
    b.title ∈ st_L.rul[uid] ▷ *Loan policy compliance*
    ⊤ == SigVeri(epk, σ_{eph}, b.title ||ts) ▷ *Valid σ_{eph}*
    ⊤ == SigVeri(π_{loan}) ▷ *Valid π_{loan}*
    then
    st_L.loan ← st_L.loan + (b.title, uid, ddl, ⊥) ▷ *Add open loan*
    c_{BB} ← Commit(b.title, ts, epk, σ_{eph}, π_{loan})
    c'_{BB} ← Append((b.title, ts, epk, σ_{eph}, π_{loan}), c_{BB})
    σ_{sk_L} ← SigSign(sk_L, b.title || ts || epk || σ_{eph} || π_{loan} || c_{BB}')
    π_{apnd} ← AppendProve(c_{BB}, c_{BB}', (b.title, ts, epk, σ_{eph}, π_{loan}))
    logBB ← logBB+(b.title, ts, epk, σ_{eph}, π_{loan}, c'_{BB}, σ_{sk_L}, π_{apnd}, h) ▷ *Append and publish the new loan record on logBB*
    logref ← #logBB[(b.title, ts, epk, σ_{eph}, π_{loan})] ▷ *Get record reference line number from logBB*
    loanres ← (logref, ddl) ▷ *Generate loan result loanres*
    return loanres, logBB, st_L.loan
  else
    return ERROR reject loan request loanreq
Reader verifies the correctness of appended loan request
procedure VerLoanResR(loanres, logBB, st_R.key, st_R.req)
  (logref, ddl) ← loanres ▷ *Parse loanres*
  assure ⊤ == SelfAuditR(logref, st_R.req) ▷ *Audit for log consistency*
  return st_R.ctdb ← (b.title, logref, ddl, ∅)

Algorithm 3 Access Protocol

User initiates access request
procedure PrepAccessU(b.title, st_U.auth, st_U.eph)
  loadreq ← (b.title, st_U.auth.uid, st_U.auth.uauth, st_U.eph.esk) ▷ *Generate access request of b.title for $\mathcal{U}$*
  ▷ *Include esk in loadreq to facilitate potentially enforced deletion and return for $\mathcal{R}$*
  return accessreq
Reader generates content loading request for user access
procedure GenLoadReqR(accessreq, logBB, st_R.ctdb, st_R.key)
  (b.title, uid, uauth, esk) ← accessreq ▷ *Parse access request*
  if
    $C(\cdot)$ ≥ st_R.ctdb[b.title].ddl ▷ *DDL elapsed*
    then
    st_R.ctdb ← st_R.ctdb − st_R.ctdb[b.title].b.cont
    return invoke GenRetReqR() of Return^{URL} with esk in input ▷ *Enforce return and deletion of content*
  assure ∃logBB[st_R.ctdb[b.title].logref] ▷ *Ensure the borrow record is on logBB*
  assure ∄logBB[st_R.ctdb[b.title].ptr] == st_R.ctdb[b.title].logref] ▷ *Ensure there is no return record for this borrow record at logref*
  if
    b.cont ∈ st_R.ctdb[b.title] and ▷ *Already downloaded*
    then return repr(st_R.ctdb [b.title].b.cont) ▷ *$\mathcal{R}$ runs repr() function to securely load b.cont from st_R.ctdb*
  loadreq ← (logref, uid, uauth, vk_{ur})
  return loadreq
Library verifies load request and returns content to reader
procedure ProcessLoadReqL(loadreq, logBB, st_L.reglst, st_L.ulst, st_L.loan)
  (logref, uid, uauth, vk_{ur}) ← loadreq
  if ▷ *Verify open loan status*
    (uid, uauth) ∈ st_L.ulst and
    (uid, vk_{ur}) ∈ st_L.reglst and
    ∃(b.title, uid, ddl, ⊥) ∈ st_L.loan and
    $C(\cdot)$ < ddl **then**
    st_L.loan ← (b.title, uid, ddl, ⊤) ▷ *Update status as loaded*
    return loadres ← b.cont
  return ERROR invalid loadreq
Reader processes content from library
procedure ProcessLoadResR(loadres, st_R.ctdb)
  b.cont ← loadres
  st_R.ctdb ← st_R.ctdb + (b.title, logref, ddl, b.cont)
  return accessres ← repr(b.cont)

Suppose $k_{\mathcal{U},\mathcal{R}} \overset{\sim}{\in} pub$ is not present in the Merkle Tree. For any such key, if $\mathcal{L}$ tries to generate a NIZK (Algorithm 2, Algorithm 2)

$$\pi_{loan} \leftarrow \text{NIZK}\{(vk_{ur}, sk_{ur}, \sigma_{ur}, \pi_{mem}, epk, \sigma_{eph}) : \ldots\}$$

for a loan request during the borrowreqUL procedure using $k_{\mathcal{U},\mathcal{R}} \overset{\sim}{\in} pub$ as vk_{ur}, the third statement in the conjunction to be proven (namely, MemberVeri(…), which verifies Merkle Tree membership) will be false. Then by the soundness of the NIZK, the proof generation procedure will return an error/⊥.

Algorithm 4 Return Procedure

```
# User initiates return request
procedure PrepReturnU(b.title, st_U.eph, st_U.auth)
    rmreq ← (b.title, st_U.eph.esk, st_U.auth.uid, st_U.auth.uauth)
    return rmreq
# Reader generates return request
procedure GenReturnReqR(rmreq, st_R.ctdb, st_R.key)
  ▷ Either invoked by 𝒰 or by ℛ when it goes online for the first time passing the return deadline ddl
    retreq ← (b.title, C(·), esk, st_R.ctdb[b.title].logref,
                  uid, uauth, st_R.key.vk_ur)
    return retreq
# Library verifies and appends return request
procedure ProcessReturnReqL(retreq, logBB, st_L.loan)
    (b.title, ts, esk, ptr, uid, uauth, vk_ur) ← retreq
    retres ← ⊥
    if
        (uid, uauth) ∈ st_L.ulst and
        (uid, vk_ur) ∈ st_L.reglst and
        ∃(b.title, uid, ddl, ∗) ∈ st_L.loan and
        SigVeri(esk, logBB [logref.σ_eph ], logBB [logref ]) == ⊤ then
        st_L.loan ← st_L.loan − (b.title, uid, ddl, ∗)
        c_BB ← Commit(b.title || ts || esk || ptr)
        c'_BB ← Append(b.title || ts || esk || ptr, c_BB)
        π_apnd ← AppendProve(c_BB, c_BB', b.title || ts || esk || ptr)
        σ_sk_L ← SigSign(sk_L, b.title || ts || esk || ptr || c_BB')
        h = H(σ_ur, σ_eph)
        logBB ← logBB + (b.title, ts, esk, ptr, c_BB, σ_sk_L, π_apnd, h)
        retres ← #logBB[(b.title, ts, esk, ptr, c_BB, σ_sk_L, π_apnd)]
    return retres, logBB, st_L.loan
# Reader verifies appended return request and removes content
procedure ProcessReturnResR(retres, st_R.ctdb)
    if
        retres == ⊥ then
        retack ← ⊥
    (b.title, ts, esk, ptr, c_BB, σ_sk_L, π_apnd) ← logBB[retres]
    if
  ▷ Self audit record consistency, variation of verilogR like Algorithm 2
        (b.title, ts, epk, σ_eph) ∉ st_R.req or
        ⊥ == SigVeri(pk_L, b.title || ts || esk || ptr || c_BB, σ_sk_L) or
        ⊥ == SigVeri(esk, b.title || ts,σ_eph) or then
        return ERROR inconsistent retres
    st_R.ctdb ← st_R.ctdb − st_R.ctdb[b.cont]      ▷ Delete book content
    retack ← ⊤
    return retack, st_R.ctdb
```

Algorithm 5 Audit Procedure

```
# Auditor verifies the correctness of all records
procedure EvalAuditProofV(tsrange, logBB, DistLst)
    if
        ∀logBB[t ∈ tsrange] :
          AppendVeri(com, com', π) == ⊤ and
          SigVeri(pk_L, σ_sk_L, b.title || ts || epk || σ_eph || π_loan || c_BB') and

          if ∃logBB[t].ptr
            SigVeri(logBB [ptr ].epk, logBB [ptr ].σ_eph, esk) == ⊤
          and |{logref : logBB[logref].ptr == t}| > 1
        and ∀b.title, DistLst[b.title] ≥ logBB[b.title]  then return ⊤
    return ⊥
# Self audit function called in Algorithm 2
function SelfAuditR(logref,st_R.req)
    (b.title, ts, epk, σ_eph, π_loan, c'_BB, σ_sk_L, π_apnd, h) ← logBB[logref]
    (vk_ur, sk_ur, π_mem) ← st_R.req
    σ_ur ← SigSign(sk_ur, b.title ||ts ||epk)
    if                                          ▷ Ensure loan record consistency
        (b.title, ts, epk, σ_eph) ∉ st_R.req or
        H(SigSign(sk_ur, b.title ||ts ||epk), σ_eph) ≠ h or
        ⊥ == NIZKVeri((vk_ur, sk_ur,σ_ur,π_mem,epk,σ_eph),π_loan) or
        ⊥ == SigVeri(pk_L, b.title || ts || epk || σ_eph || π_loan || c_BB', σ_sk_L)
        then
        return ERROR invalid loanres
    return ⊤
```

Suppose $k^{\sim\ pub}_{\mathcal{U},\mathcal{R}}$ is present in the Merkle Tree. Then, due to the unforgeability of the signature scheme, the corresponding $k^{\sim\ priv}_{\mathcal{U},\mathcal{R}}$ cannot be a valid signing key for this verification key $k^{\sim\ pub}_{\mathcal{U},\mathcal{R}}$.

The reduction to unforgeability is as follows: Suppose, for contradiction, that $k^{\sim\ priv}_{\mathcal{U},\mathcal{R}}$ is a valid signing key for the verification key $k^{\sim\ pub}_{\mathcal{U},\mathcal{R}}$. We design an adversary that wins the signature unforgeability game, as follows: first, given the target verification key vk, create a Merkle Tree containing vk as a registered patron key according to LendLocked (simulating all parties involved), and give this to $\tilde{\mathcal{U}}$. Given $\tilde{\mathcal{U}}$'s response $k^{\sim\ pub}_{\mathcal{U},\mathcal{R}} = (vk, k^{\sim\ priv}_{\mathcal{U},\mathcal{R}})$, the adversary chooses a random message string m and sign m with $k^{\sim\ priv}_{\mathcal{U},\mathcal{R}}$, outputting the resultant signature along with m. Whenever $\tilde{\mathcal{U}}$ succeeds at outputting this signature key pair, this adversary wins the unforgeability game. Hence, we see $\tilde{\mathcal{U}}$ may not produce a valid signing key with greater than negligible probability.

Returning Correctness (Def. 9). Recall that returning correctness requires that after the return algorithm returndelURL has been run honestly for a user $\mathcal{U}$ with id uid and book title b.title, that user $\mathcal{U}$ does not have an open loan on the book b.title. As part of returndelURL, $\mathcal{L}$ runs the algorithm ProcessReturnReqL (Algorithm 4), which appends to logBB a log entry as follows:

$$(\text{b.title}, \text{ts}, \text{esk}, \text{ptr}, c_{\text{BB}}, \sigma_{\text{sk}_L}, \pi_{\text{apnd}}, h),$$

which, by construction of Algorithm 4, is a valid return log entry with respect to $\mathcal{U}$ and b.title. Then, by the definition of open loan (Def. 8), which requires a catalog state containing a borrow entry without a corresponding return entry, it follows immediately that $\mathcal{U}$ does not have an open loan for b.title.

Returning Soundness (Def. 10). This property follows from returning correctness (Def. 9) and borrowing and reading soundness (Definitions 7 and 12). Consider any malicious PPT algorithm $\mathcal{U}$ with id uid and keys $\text{k}_u, \text{k}_{ur}$, and any book b.title, index i, and bulletin logBB. Suppose this user is not registered with the library. Then borrowing soundness (Def. 7) implies that any borrow request by that user will be unsuccessful. Then, by the definition of open loan (Def. 8), which requires a successful borrow request recorded on the catalog, this user cannot have any open loans. Now suppose

instead that the user *is* registered with the library. Immediately after returndelURL is run (Eq. (4), Def. 10), then, returning correctness (Def. 9) implies that openloan(logBB′, uid, b.title) = 0. Thus, whether or not the user is a registered patron with the library, we have openloan(logBB′, uid, b.title) = 0.

Then, reading soundness (Def. 12) immediately implies

$$\Pr\left[\mathsf{b.cont}[i] = \tau \middle| \mathsf{readUR}(\mathsf{uid}, \mathsf{k}_u, \mathsf{b.title}, i, t) = \tau\right] \leq \mathsf{ngl}(\lambda), \quad (10)$$

exactly as required.

Borrowing and Returning Privacy (Def. 11). Consider any borrow request

$$\mathsf{borrowreqUL}(\mathsf{uid}, \mathsf{k}_u, k_{\mathcal{U},\mathcal{R}}, \mathsf{b.title})$$

at time ts and any return request

$$\mathsf{returndelURL}(\mathsf{uid}, \mathsf{k}_u, k_{\mathcal{U},\mathcal{R}}, \mathsf{b.title})$$

at time $ts' > ts$, and let $\mathsf{logBB}_0, \mathsf{logBB}_1$ be the partial catalog states as defined in Def. 11. Let ρ, ρ' be the resulting log entries. The information (b.title, ts) that the simulator gets in Def. 11 is information that is public about that borrow request on logBB_0, by construction of LendLocked (see Algorithm 2, Algorithm 2 and Algorithm 4, Algorithm 4). The remaining (non-public) information that is input to these borrow and return requests consists of a user id and user keys. We design our simulator to generate a fake pair of borrow and return entries $\varepsilon, \varepsilon'$ using an arbitrary user id and random keys generated with the honest key generation algorithms for users, including ephemeral keys.

If the simulator instead receives a borrow and return request, the simulator will instead generate a fake pair of borrow and return keys $\varepsilon, \varepsilon'$ using an arbitrary user id and random keys generated with the honest key generation algorithms for users, as before. Now, it will append to logBB_0 first the log corresponding to (b.title, ts) using the generated keys ε, followed by logBB_1, and finally followed by the return log corresponding to (b.title, ts') using ε'.

For the same reason as above, this is distributed exactly as logBB. □

J.2 Proving Read Functionality

Lemma 3. *The constructions of patronregUL, borrowreqUL, readUR, and trackreadR in Section 6 realize the read functionality. That is, they satisfy Definition 4.*

Proof. We have already proven borrowing and reading correctness (see proof of Theorem 2). It remains to prove reading soundness and privacy (Definitions 12 and 13), as follows.

Reading Soundness (Def. 12). Consider any malicious PPT algorithm $\mathcal{U}$ and honest reader $\mathcal{R}$ and any book b.title, and suppose that $\mathcal{U}$ does not have an open loan for b.title., By construction, GenLoadReqR() (Algorithm 3) requires $\mathcal{R}$ to output $\perp$ unless (a) there exists a borrow record for $\mathcal{U}$ and b.title on logBB (Algorithm 3, Algorithm 3) and (b) there does not exist a corresponding return record on logBB (Algorithm 3, Algorithm 3). Recall that $\mathcal{R}$ is treated as honest in this context (Table 2),[23] By the definition of open loan (Def. 8), and our assumption that $\mathcal{U}$ does not have an open loan on b.title, it must hold that for any valid borrow record for $\mathcal{U}$ and b.title on logBB, there must be a corresponding return record on logBB. Therefore, the condition "(a) and (b)" must be false, so GenLoadReqR() (which implements readUR) will return $\perp$, satisfying (5).

Reading Privacy (Def. 13). By construction, no subprocedures of readUR (Algorithm 3) involve any party other than $\mathcal{U}$ and $\mathcal{R}$. Recall from § 5 that user-reader pairs interact by direct physical connection. Moreover, no subprocedures of readUR queries alter the bulletin board. Therefore, all inputs to the adversary are the same in both probability expressions in Def. 13, so the adversary can have no distinguishing advantage. □

We note that the analysis of reading privacy above is trivial, and not cryptographic. See Section G.2 for discussion of why we define this property despite this seeming triviality. We include this simple analysis of why reading privacy holds, for the sake of completeness.

J.3 Proving Audit Functionality

Lemma 4. *The constructions of auditlogV, verilogR, and addlogURL in Section 6 realize the auditing functionality. That is, they satisfy Definition 5.*

Proof. We must prove correctness (Def. 14), soundness (Def. 15), and privacy (Def. 18) of auditing, which are addressed in turn below.

Auditing Correctness (Def. 14). Let $k, \lambda \in \mathbb{N}$, and let

$$\{(\mathsf{ts}_i, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)\}_{i\in[k]}$$

be a sequence of k credentials, and let logBB be the bulletin formed by appending $\mathsf{addlogURL}(\mathsf{ts}_i, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)$ in sequence.

For public-auditing correctness (Eq. (6)): Recall that by construction, auditlogV (instantiated by EvalAuditProofV() in Algorithm 5) checks that:

(a) the logBB is append only since the last audit (Algorithm 5, Algorithm 5),
(b) $\mathcal{L}$ has signed each record (Algorithm 5, Algorithm 5),
(c) the ephemeral signature in the log entry verifies successfully (Algorithm 5, Algorithm 5),
(d) no borrow record is referenced by more than one return record (Algorithm 5, Algorithm 5), and
(e) that no over-loaning is occurring (with respect to a stated over-loaning policy DistLst) (Algorithm 5, Algorithm 5).

If all these conditions hold, EvalAuditProofV() accepts; otherwise, it rejects (Algorithm 5). The series of log entries making up logBB in Def. 14 is append-only by construction, satisfying (a). In the honest execution of addlogURL, the library signature and the ephemeral signature are produced honestly according to the Borrow and Return algorithms (Algorithms 2 and 4), so by the correctness of the signature scheme, (b) and (c) hold with overwhelming probability. Finally, by assumption in Def. 14, no borrow record is referenced by more than one return record and logBB satisfies the overloaning policy DistLst, so (d) and (e) also hold. Therefore, EvalAuditProofV() will accept with overwhelming probability as required by Eq. (6).

For self-auditing correctness (Eq. (6)): Recall that verilogR (implemented by SelfAuditR in Algorithm 5) checks four conditions: that a record matches the checking e-reader's internal state , that the hash within a given record matches the hash function output on the e-reader's private inputs , that the NIZK and the signature

[23] See § 5.3 for discussion on why we treat $\mathcal{R}$ as honest here.

associated with a given record verify (Algorithm 5,). If all these conditions hold, SelfAuditR accepts; otherwise, it rejects (Algorithm 5). Since Def. 14 assumes the log entries are generated honestly using addlogURL, by the correctness of the signature scheme and the NIZK, and the fact that the hash function is deterministic, the latter three conditions are satisfied. Moreover, since Def. 14 assumes the e-reader $\mathcal{R}$ with the relevant credentials is involved in each addlogURL protocol, and the e-readers are treated as honest for this definition, the first condition holds too. Therefore, SelfAuditR will accept with overwhelming probability as required by Eq. (7).

Auditing Soundness (Def. 15). It remains to prove self-auditing soundness (Definition 16) and log collision-resistance (Definition 17). We prove these below.

Self-Auditing Soundness (Def. 16). Let $k, \lambda \in \mathbb{N}$, and let

$$\{(\mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)\}_{i\in[k]}$$

be a sequence of k credentials, and let logBB be the bulletin formed by appending the output of

$$\mathsf{addlogURL}(\mathsf{ts}, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)$$

in sequence for $i \in [k]$. Let $\mathsf{k}_{ur,i} = (\mathsf{epk}_i, \cdot, \mathsf{vk}_{\mathsf{ur},i}, \cdot)$. Let $\mathcal{R}$ be an e-reader for user $\mathcal{U}$ with id uid′. Let $(\mathsf{ts}', \mathsf{uid}', \mathsf{b.title}', \mathsf{k}'_{ur})$ be any efficiently computable tuple that is not in logBB (as in Eq. (8)) and is the sole entry in $\mathcal{R}$'s state st_R.req. Let $\mathsf{k}'_{ur} = (\mathsf{epk}', \mathsf{esk}', \mathsf{sk}'_{\mathsf{ur}}, \mathsf{vk}'_{\mathsf{ur}})$.

Now suppose, for contradiction of Def. 16, Eq. (9), that there is some logref such that verilogR(logref) = ⊤ with non-negligible probability when run by $\mathcal{R}$. Let

$$\mathsf{logBB[logref]} = \mathsf{addlogURL}(\mathsf{ts}_i, \mathsf{uid}_i, \mathsf{b.title}_i, \mathsf{k}_{ur,i}, \sigma_i)$$

for some index i where $\mathsf{k}_{ur,i} = (\mathsf{epk}_i, \mathsf{esk}_i, \mathsf{sk}_{\mathsf{ur},i}, \mathsf{vk}_{\mathsf{ur},i})$. In each log entry, a timestamp ts, a title b.title, and an ephemeral public key epk are in the clear. By definition of verilogR (Algorithm 5), and by our assumption on st_R.req, verilogR(logref) = ⊤ (assumed earlier) implies that logBB[logref] contains ts′, b.title′ and epk′ in the clear, that is, $\mathsf{ts}' = \mathsf{ts}_i$ and $\mathsf{b.title}' = \mathsf{b.title}_i$ and $\mathsf{epk}' = \mathsf{epk}_i$.

Because $(\mathsf{uid}', \mathsf{b.title}', \mathsf{k}'_{ur})$ is not in logBB by assumption, it must hold that

$$(\mathsf{b.title}', (\mathsf{epk}', \mathsf{vk}'_{\mathsf{ur}})) \neq (\mathsf{b.title}_i, (\mathsf{epk}_i, \mathsf{vk}_{\mathsf{ur},i})). \tag{11}$$

Since we already established $\mathsf{b.title}' = \mathsf{b.title}_i$ and $\mathsf{epk}' = \mathsf{epk}_i$, for Eq. (11) to hold, it must hold that $\mathsf{vk}'_{\mathsf{ur}} \neq \mathsf{vk}_{\mathsf{ur},i}$.

Let $\pi_{\mathsf{loan},i}$ be the NIZK proof in logBB[logref]. By construction, $\pi_{\mathsf{loan},i}$ proves that some signature σ^* on message $(\mathsf{b.title}_i, \mathsf{ts}_i, \mathsf{epk}_i)$ is valid with respect to verification key $\mathsf{vk}_{\mathsf{ur},i}$ (Algorithm 2, Algorithm 2). However, the NIZK verification in verilogR(logref) = ⊤ (assumed earlier) implies that $\pi_{\mathsf{loan},i}$ is also a valid NIZK proof that the same signature σ^* on the same message

$$(\mathsf{b.title}_i, \mathsf{ts}_i, \mathsf{epk}_i) = (\mathsf{b.title}', \mathsf{ts}', \mathsf{epk}')$$

is valid with respect to $\mathsf{vk}'_{\mathsf{ur}}$. By soundness of the NIZK, it then holds with overwhelming probability that σ^* is a valid signature with respect to both verification keys $\mathsf{vk}_{\mathsf{ur},i}$ and $\mathsf{vk}'_{\mathsf{ur}}$, which we previously concluded were not equal.

Finally, recall that $\mathsf{vk}'_{\mathsf{ur}}$ is efficiently computable, but our signature scheme's collision resistance property implies that it is computationally hard to find distinct verification keys such that σ^* that verifies with respect to both keys, so we have a contradiction.

Log Collision Resistance (Def. 17). Consider the collision-resistance game in Def. 17. For the sake of contradiction, now consider honestly generated keys $\mathsf{k}_{ur}, \mathsf{k}'_{ur}$ and corresponding log entries

$$\mathsf{addlogURL}(\mathsf{uid}, \mathsf{b.title}, \mathsf{k}_{ur}, \sigma) \neq \bot,$$
$$\mathsf{addlogURL}(\mathsf{uid}', \mathsf{b.title}', \mathsf{k}'_{ur}, \sigma') \neq \bot$$

with uid ≠ uid′ and b.title ≠ b.title′.

By definition of the borrowreqUL and returndelURL procedures (Algorithms 2 and 4, respectively), valid entries[24] in addlogURL include a commitment H of the signature (σ or σ' above) using the user-reader pair's key. Because the signature is binding, signatures of H are also collision-resistant. Then by collision resistance, with overwhelming probability, $\sigma = \sigma'$. Because the keys k_{ur} and k'_{ur} are generated independently, with overwhelming probability they are not equal. The unforgeability of the signature scheme implies that for any messages m, m' and honestly generated keys $\mathsf{k}_{ur} \neq \mathsf{k}'_{ur}$, the probability that a signature σ on message m honestly generated using k_{ur} verifies successfully with respect to the other key-message pair (k'_{ur}, m') is negligible. (Otherwise, an adversary could efficiently win the signature forgery game as follows: (1) honestly generate a key pair and use it to produce a signature σ on m, (2) check if σ also verifies successfully with respect to the challenge key on m'; (3) if yes, output (σ, m'), otherwise, go back to step 1.)

Thus, we have a contradiction.

Auditing Privacy (Def. 18). Let $\mathcal{A}$ be PPT. To prove auditing privacy, we must design a simulator Sim which takes in the timestamps and book titles $info$ of a given logBB and outputs logBB′ such that $\mathcal{A}(\mathsf{logBB}) \approx \mathcal{A}(\mathsf{logBB}'$. Our Sim will function as so: First, Sim emulates $\mathcal{L}$ and generates its own signing key $signskl^{\mathsf{Sim}}$. On input $info$ for $info = \{t_i, \mathsf{b.title}_i\}_{i\in[|\mathsf{logBB}|]}$, for each $i = 1, \ldots, k$, Sim samples a random user key $\mathsf{epk}^{(i)}$ and user-reader signature $\sigma^{(i)}_{\mathsf{eph}}$ (with associated secret key and signing key, respectively). Sim then generates π_{loan}, π_{apnd}, and h as in the borrowreqUL procedure with these keys and appends

$$(\mathsf{b.title}_i, \mathsf{ts}, \mathsf{epk}^{(i)}, \sigma^{(i)}_{\mathsf{eph}}, \pi_{\mathsf{loan}}, c'_{\mathsf{BB}}, \sigma_{\mathsf{sk}_L}, \pi_{\mathsf{apnd}}, h),$$

as in the borrowreqUL protocol. If $\mathsf{b.title}_i = \mathsf{b.title}_j$ for some $j < i$ and $t_i > t_j$, Sim will instead process this as a return log, and will sample the same keys as before but instead append a log as in the returndelURL protocol. Finally, Sim outputs the simulated logBB′.

Because the simulator generates its keys as if it was a user, the simulator's log logBB′ is indistinguishable to the real logBB for any party who does not know the actual secret and signing keys, as semantic security of the encryption signature scheme ensures each ciphertext can be swapped from the real to the simulated log. The adversary is not given these keys (only the complete log). So, reducing to semantic security, with overwhelming probability $\mathcal{A}$ cannot distinguish between logBB′ and logBB. □

J.4 Proving Copy Deterrence

Lemma 5. *The constructions of readUR and trackreadR in Section 6 satisfies copy deterrence (Definition 19).*

[24] Valid entries are those that are not equal to ⊥.

Proof. Let b.title be arbitrary, let $n = |\text{b.cont}[\cdot]|$, and let m be the length to which $\mathcal{R}$ pads each entry in b.cont$[\cdot]$. We must construct a simulator such that no efficient adversary can distinguish between the real and ideal ensembles. Our simulator Sim will work as follows: On startup, Sim samples n keys $k_1, \ldots, k_n$. according to the key generation algorithm of the symmetric key cryptosystem that $\mathcal{R}$ uses (i.e., KDF). Sim proceeds to encrypt 0^m with each key k_i using the cryptosystem. We denote each $c_i = \mathsf{Enc}(0^m; k_i)$.

From this, Sim initializes its dump response to $\{c_i\}_{i \in [n]}$. Whenever Sim receives its first forwarded query from $\mathcal{F}$ for readUR(i), Sim sets a flag to i and decrypts c_i using k_i. On any following queries, Sim encrypts the last flagged entry and then does the above.

Note that Sim acts exactly as $\mathcal{R}$ does, except instead of performing trackreadR for b.title, it does so for a book that is all 0's (and the same length as b.title). By the semantic security of the encryption scheme, the adversary cannot distinguish as each real ciphertext is swapped for the corresponding simulated ciphertext. So, reducing to semantic security, with overwhelming probability, no efficient adversary can distinguish between the real and simulated ensembles. □