Building Privacy-Preserving Cryptographic Credentials from Federated Online Identities

John Maheswaran
Yale University
john.maheswaran@yale.edu

Daniel Jackowitz
Yale University
daniel.jackowitz@yale.edu

Ennan Zhai
Yale University
ennan.zhai@yale.edu

David Isaac Wolinsky
Yale University
david.wolinsky@yale.edu

Bryan Ford
Swiss Federal Institute of Technology (EPFL)
bryan.ford@epfl.ch

ABSTRACT

Federated identity providers, *e.g.*, Facebook and PayPal, offer a convenient means for authenticating users to third-party applications. Unfortunately such cross-site authentications carry privacy and tracking risks. For example, federated identity providers can learn what applications users are accessing; meanwhile, the applications can know the users' identities in reality.

This paper presents Crypto-Book, an anonymizing layer enabling federated identity authentications while preventing these risks. Crypto-Book uses a set of independently managed servers that employ a (t, n)-threshold cryptosystem to collectively assign credentials to each federated identity (in the form of either a public/private keypair or blinded signed messages). With the credentials in hand, clients can then leverage anonymous authentication techniques such as linkable ring signatures or partially blind signatures to log into third-party applications in an anonymous yet accountable way.

We have implemented a prototype of Crypto-Book and demonstrated its use with three applications: a Wiki system, an anonymous group communication system, and a whistleblower submission system. Crypto-Book is practical and has low overhead: in a deployment within our research group, Crypto-Book group authentication took 1.607s end-to-end, an overhead of 1.2s compared to traditional non-privacy-preserving federated authentication.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General | Security and protection

1. INTRODUCTION

Federated identity services have gained widespread popularity among users as a simplifying means for managing their online identities. In particular, federated identity providers (*e.g.*, Facebook and PayPal) employ authentication protocols such as OAuth [20, 21] and OpenID [34] to offer their users the convenience of using single identity – and thus a single credential – to log into and access various third-party applications (*e.g.*, Wiki and StackOverflow). This convenience, however, comes at the privacy cost of enabling not only the federated identity providers themselves, but also the third-party application providers and advertising partners, to link a user's account and track her activities across applications.

In order to tackle the above privacy problem, this paper presents Crypto-Book, an architecture that enables users to log into third-party applications anonymously or pseudonymously using their existing federated identities. At a high level, Crypto-Book could be looked as an anonymizing layer between the user's federated identity providers and third-party applications consuming the identities. This anonymizing layer generates cryptographic pseudonyms for users based on their federated identities, thus enabling users to anonymously log into third-party applications with the pseudonyms. Such pseudonyms not only prevent the application providers from learning the user's actual identity/profile information, but also blocks the federated identity providers from linking a user's identity across different applications. Furthermore, Crypto-Book is capable of generating cryptographic pseudonyms from a combination of multiple federated identities (*e.g.*, both Facebook and PayPal identities) to remain secure in the event that one of user's federated identity provider accounts is compromised.

While there have been many cryptographic schemes for anonymous authentication [1–3, 5, 22, 23], these efforts typically assume that users have public keys that are known or validated through PKI – an assumption that has proven a major roadblock to widespread use [40]. On the contrary, Crypto-Book aims at providing a systematic effort that can incorporate different privacy-preserving cryptographic authentication methods without requiring any conventional PKI or PGP-style key management.

Crypto-Book's architecture contains three logical components: 1) a given *client* that represents the user; 2) a group of *credential producers* that interact with the client to manufacture cryptographic credentials for the client; and 3) *credential consumers* that validate these credentials and use them to create application-specific pseudonymous accounts for the client. The credential producers, each maintained or run individually, verify the user's claimed identity via the federated identity providers, then jointly manufacture privacy-preserving credentials, so that the user does not need to trust any one (or few) of these credential producers. While credential production itself uses standard threshold cryptography, a key systems challenge Crypto-Book addresses is ensuring that *all* of the credential producers can securely and independently authenticate the user's federated identity via the unmodified OAuth protocol, without demanding that the human user perform multiple tedious and seemingly-redundant OAuth logins in succession.

Users' privacy expectations, in practice, often depend on the third-party applications the user is visiting. The Crypto-Book ar-

CODASPY'16, March 9–11, 2016, New Orleans, LA, USA.
© 2016 ACM. ISBN 978-1-4503-3935-3/16/03...$15.00
DOI: http://dx.doi.org/10.1145/2857705.2857736

chitecture is therefore designed to support different forms of privacy protection through multiple "pluggable" cryptographic credential schemes. As examples of such designs, this paper presents and implements two distinct credential schemes within the Crypto-Book architecture.

- Crypto-Book can use partially blind signatures [1, 9] to create "at-large" anonymous access tokens enabling users to log into third-party applications, optionally revealing only *user-selected* information about the user to those applications, while protecting the user's identity from disclosure or linking. For example, a partially-blind credential could indicate that the user is over 21 years of age and has had a PayPal account in good standing for at least a year (detailed in §4).
- Using group credentials built on linkable ring signatures [26,35], Crypto-Book enables users to prove to be one of a set of federated identities (*e.g.* defined by a list of Facebook identities) without disclosing which member of the set they are, and **without requiring the *other* listed members to be Crypto-Book users or have cryptographic keypairs (detailed in §5)**.

These two pluggable credential schemes are intended only as a useful starting point and are not intended to be definitive: they could be improved in many ways utilizing more advanced cryptographic authentication techniques [38].

Our prototype implements these two credential schemes and supports both PayPal and Facebook as federated identity providers. We also prototyped three applications to evaluate Crypto-Book: 1) a Wiki system exploring the use-case of supporting anonymous but accountable editing on sites like Wiki; 2) an accountable anonymous chatting system, Dissent [41], offering scalable and traffic analysis resistant communication service; and 3) a SecureDrop [36] like application enabling a whistleblower to convince a journalist that she is a member of some authoritative group – such as a government official or company employee of a certain rank – without revealing her precise identity. We have deployed Crypto-Book login for a web site used within our research group and evaluated the overall login time, as well as various performance microbenchmarks for each stage of the credential pickup and login processes.

To summary, this paper makes the following contributions: 1) a practical architecture offering privacy-protected and accountable usage of existing federated identities; 2) multiple pluggable credential schemes supporting different degrees of privacy and anonymity; 3) credentials derived from multiple federated identities, increasing protection against identity compromise; and 4) comprehensive experiments demonstrating the practicality of the system.

2. BACKGROUND AND MOTIVATION

This section first describes privacy issues of federated identity services (§2.1), and then presents three motivating use-cases for Crypto-Book (§2.2).

2.1 Privacy Concerns with Federated Identity

Through federated identity based authentication, a user can log into third-party applications without having to maintain separate accounts for each of the applications. Well-known federated identity providers in practice include Facebook, Google+ and PayPal; representative third-party applications supporting federated identity authentication include Quora and StackOverflow. While federated authentication offers great convenience from the perspective of the user, it also introduces or exacerbates several privacy risks, of which Crypto-Book focuses on the following.

- The federated identity providers can learn every application or site the user logs into using her federated identity, and *every time* that they do so.
- Third-party applications can learn the user's true identity and often many profile details such as friends lists and location.
- Third-party applications can link users, and their corresponding profile details, across applications, sharing or selling the information to advertisers.
- If one of a user's federated identity accounts is compromised, the attacker is then able to access third-party applications using this account.

Furthermore, whenever a user visits any page containing federated identity providers' "Like" or "Share" buttons, the identity providers may again learn that the user visited that page, enabling even more detailed tracking and sale of personal information. Additionally, third-party applications often demand access to profile information, contacts lists, and even write access (permission to post on user's behalf) and it is sometimes unclear to users what these permissions will be used for, and not obvious after-the-fact how they were actually used.

2.2 Motivating Use-Cases for Crypto-Book

We now present three motivating use-cases for Crypto-Book. For each use-case, we have built and evaluated a representative application using the Crypto-Book framework.

Privacy-preserving "Login with Crypto-Book". Crypto-Book can be used to provide general, privacy-preserving login functionality to third-party applications. An application may choose to include a "Login with Crypto-Book" button which allows users to be authenticated via Crypto-Book. The difference between Crypto-Book privacy preserving login and existing federated authentication, *e.g.*"Login with Facebook", is that Crypto-Book login preserves the user's privacy and anonymity, while also protecting the third-party applications from abuse. The application learns that the user has been authenticated by Crypto-Book, and learns a pseudonym for the user, but does not know the user's actual identity, nor can the application map back from the pseudonym to the user's identity. For example, while Wiki allows anonymous editing, such privileges are often abused for vandalism. We built a system, CB-Wiki, that allows users to edit pages without revealing their identities but at the same time allows the Wiki site administrators to sanction site abusers. CB-Wiki leverages Crypto-Book to provide anonymous yet linkable editing in a Wiki system environment.

Abuse-resistant anonymous communications. Crypto-Book is additionally useful for authenticating users within anonymous chat applications. Organizations may wish to allow members to discuss sensitive issues without revealing their individual identities but at the same time limit access to their discussions to only members of the organization in an effort to prevent repressive authorities or other undesirable outsiders from viewing the communications. The Crypto-Book architecture could be used to give such applications a reason not to block Tor [16], by authenticating users anonymously in an abuse resistant manner. This would allow applications to counter anonymous abuse without compromising anonymity. The Tor Project issued a "call to arms" seeking solutions to this issue[1]. We built CB-Dissent on Dissent [12,41], an anonymous group communication application. CB-Dissent shows how the integration of anonymous authentication with anonymous communication systems can better protect the identities of the users.

[1]https://blog.torproject.org/blog/call-arms-helping-internet-services-accept-anonymous-users

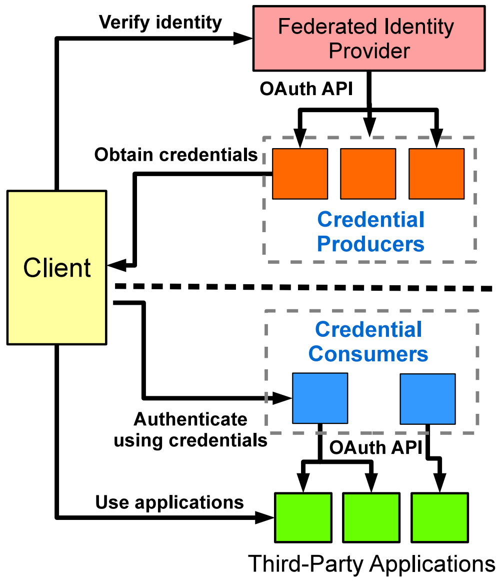

Figure 1: The Crypto-Book architecture overview. Crypto-Book consists of credential producers and credential consumers. Conventional federated authentication process only contains client, federated identity provider and third-party applications.

Group authenticated whistleblowing. Crypto-Book authentication supports verifiable whistleblowing by allowing a journalist taking possession of sensitive documents to authenticate the documents without compromising the anonymity of the source. To illustrate this usage model, we built CB-Drop by extending the SecureDrop [36] open-source whistleblowing platform. CB-Drop is able to provide anonymous document signing using Crypto-Book identities, allowing for verifiable leaks without compromising privacy. A whistleblower authenticates as a member of a group, so that a journalist, for example, can verify that the leak comes from one of the members of the ring, yet does not know which specific member leaked the document.

3. OVERVIEW

This section first presents a high-level overview of the Crypto-Book architecture (§3.1). We then present Crypto-Book's privacy goals (§3.2) and define our assumptions (§3.3). Finally, we detail two key components of Crypto-Book: credential producers (§3.4) and credential consumers (§3.5).

The purpose of this section is to show a basic working process of the Crypto-Book architecture. To highlight Crypto-Book as a "pluggable" architecture, the following two sections (§4 and §5) each detail a specific credential scheme that has been plugged into the Crypto-Book architecture, respectively.

3.1 Crypto-Book Architecture Overview

Figure 1 shows the overall Crypto-Book architecture. In the conventional scenario (*i.e.*, without Crypto-Book), a federated identity provider exposes an authentication API to third-party applications (*e.g.*, via OAuth [21]), which enables users to log into these applications with their federated identities. Such protocols, however, may expose users' identities or profile information to the third-party applications and permit linkage across applications (as mentioned in §2.1).

Crypto-Book addresses this concern by interposing two additional, disjoint layers – *credential producers* and *credential consumers* – between the federated identity providers and the third-party applications. In particular, *credential producers* interact with the federated identity providers to collectively map clients' federated identities to privacy-preserving cryptographic credentials. Clients then submit these credentials to *credential consumers*, which create cryptographic pseudonyms/accounts for the clients, allowing the clients to authenticate with cooperating third-party applications using these pseudonyms rather than their true federated identities.

3.2 Privacy Goals

Crypto-Book targets the following privacy goals: *anonymity*, *unlinkability*, and *accountability*.

- Anonymity means that no party can associate any operation within a third-party application with a specific client's federated identity; additionally, federated identity providers cannot learn what third-party applications a specific client has accessed.
- Unlinkability means that no party can link or track any client's pseudonymous identities across multiple third-party application providers.
- Accountability means pseudonymous identities who abuse applications can be identified and held accountable (*e.g.*banned) without revealing the corresponding federated identity.

Non-goals. Because Crypto-Book aims to preserve the anonymity of clients, defending against sybil attack [17] and network-level Denial-of-Service (DoS) attack have been out of our scope. These attacks are important in practice, but not specific to Crypto-Book. Well-known defenses for these two attacks could be applied.

3.3 Threat Model and Assumptions

Clients are potentially malicious in that they may abuse third-party applications, *e.g.*, posting low-quality content on Wiki. We do not consider network level attacks and assume that clients are able to connect to system components through an anonymous network such as Tor [16].

Both *federated identity providers* and *third-party application providers* are potentially malicious in that they try to break the privacy properties described in §3.2 – for example, de-anonymizing clients' data and linking some client to a specific federated identity. In addition, multiple application providers are allowed to collude with each other. Collusive application providers may try to track a client's identity across their applications. Federated identity providers can collude with application providers, and they may try to learn what applications a specific client is accessing or has accessed.

We assume that fewer than t of n ($t \leq n-1$) credential producers are dishonest, and may collude with each other to attempt to forge user credentials. The remaining producers are honest-but-curious. In particular, these producers faithfully follow their protocols, but may try to exploit additional information that can be learned in doing so.

We assume that all the cryptographic primitives are operated correctly and work securely. We assume anonymous network communication (*e.g.*, Tor), linkable ring signatures and blind signatures cannot be compromised.

3.4 Credential Producers

Crypto-Book credential producers are a set of independently managed servers responsible for producing cryptographic credentials from verified federated identities. We assume each server is run by a respected, technically competent, and administratively independent anonymity service provider. We envision several commercial

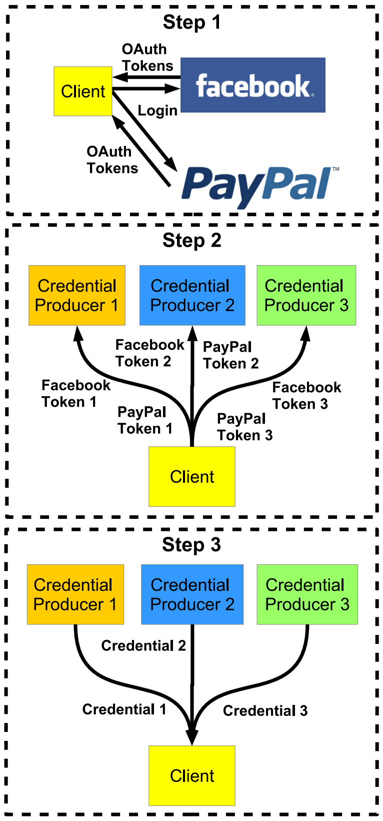

Figure 2: Client collects credentials from multiple credential producers.

or non-profit organizations each deploying a cluster of the credential producer servers, as either a for-profit or donation-funded community service.This assumption on servers has been demonstrated to be reasonable in practice [16, 41]. These servers can support various credential schemes (shown in §4 and §5), enabling Crypto-Book to satisfy different users' privacy expectations in practice.

To obtain cryptographic credentials, a Crypto-Book client contacts a threshold t of the n credential producers, each of which independently authenticates the user with respect to one or more federated identity providers. This process is outlined in Figure 2, and proceeds as the following three steps. **We note that credential production is performed only once for each third-party application that a client wishes to access, not on every authentication.**

Step 1. Each credential producer prompts the user to perform a non-anonymous OAuth federated authentication with each of the federated identity providers (*e.g.*, Facebook and PayPal). Each credential producer then redirects the client to the federated identity provider's login page for authentication. Upon successful authentication, the client receives a unique OAuth token corresponding to the specific federated identity provider and identity.

Step 2. The client sends these tokens to the producer who initiated that authentication. With these tokens, the corresponding credential producers can access and validate the user's profile information. Credential producers request only the minimum access necessary to verify that the identity is valid. Each credential producer verifies via federated identity provider's profile-access API (*e.g.*, the Facebook API) that the federated identity for which the OAuth token was obtained corresponds to the federated identity (*e.g.*, Facebook ID) that the user claimed to have. For multiple federated identity providers (*e.g.*, Facebook and PayPal), each credential producer also verifies that the user attributes (*i.e.*, date of birth and email address) are the same for both the Facebook and PayPal accounts. In order to obtain a verified PayPal account, a user needs to connect her real-world bank account or credit card, which requires showing her real-world identity (*e.g.*, driver license) in person at a bank. **This provides a higher barrier to entry and makes it much more difficult for someone to assume a fake identity.**

Step 3. After each producer verifies all identities with their respective providers, assuming all verify successfully, the producer returns a share of the cryptographic credential to the client. The client then combines the shares from all contacted producers and stores the resulting cryptographic credential for use in future privacy-protected logins.

3.4.1 Design Intuitions

It is crucial that each producer performs its own OAuth authentication and receives its own OAuth token. A strawman design uses only one OAuth workflow with a single OAuth token and forwards this token to each of the credential producers. The problem with this is that each credential producer can forward the token to other producers to impersonate the user. Having separate OAuth workflows for each producer protects against this.

While security requires each of multiple credential producers to verify one or more of the user's federated identities, we do not wish to subject the *human* user to a tedious process of typing passwords into many federated identity provider login dialogs in succession. The Crypto-Book client hides the multiple-independent-authentications from the human user, on the client side using a Chrome plugin.

Crypto-Book credential producers support multiple cryptographic credentials, requiring only that the credential be adaptable to a (t, n)-threshold cryptosystem; any set of at least t honest producers must be able to produce a valid credential which will be accepted by any honest credential consumer, while any set of fewer than t producers must not be able to produce such a credential.

3.5 Credential Consumers

Credential consumers map cryptographic credentials to cryptographic pseudonyms that can then be used to authenticate with third-party applications. The cryptographic pseudonyms produced by credential consumers are unlinkable to the actual federated identities from which they are derived. Credential consumers typically take one of two forms: OAuth provider or application-embedded consumer.

OAuth provider consumer. OAuth provider consumers operate externally to third-party applications. They map cryptographic credentials to anonymous identities and then expose those identities to third-party applications via the OAuth protocol. Applications interact with OAuth provider consumers just as they would directly with conventional federated identity providers. Third-party applications already using federated authentication require little modification to support OAuth provider credential consumers; such integration, however, requires that the application trust the OAuth provider. There is nothing, however, preventing a third-party application from running its own OAuth provider consumer, independent of the application but still under the same administrative domain.

Application-embedded consumer. An application-embedded credential consumer exists directly within a third-party application, either via an imported library or a custom implementation of the consumer. Using this approach the application need not trust an external provider, but at the cost of ease of integration with existing authentication mechanisms.

4. AT-LARGE CREDENTIAL SCHEME

This section presents the first of two concrete credential schemes we have built on top of Crypto-Book. The *at-large credential scheme* does not explicitly constrain the anonymity set of a user, but instead represents that the user has been verified as the owner of *some* federated identity. The anonymity set for each at-large credential is then implicitly all users who have ever collected a credential in the time period before the credential was used. Credential producers enforce the accountability of at-large credentials by restricting the number of credentials they produce for a given federated identity within a period of time (*i.e.* rate-limiting). We then extend at-large credentials to optionally include *credential attributes*, such as "age over 18" or "identity active for at least one year", which are also verified with the identity provider at the time of credential production.

4.1 Building Block: Blind Signatures

The blind signature [1, 9, 37] is a cryptographic primitive, where a *requester* can request a *signer* to sign one or more messages, while the signer cannot learn the signed message's content. Given the message-signature pair, a public *verifier* is able to verify the legitimacy of the signature. In our architecture, the client is the requester, each credential producer is a signer and the credential consumer is a verifier (of multiple signatures). In our work we use the blind signature scheme proposed by Shen *et al.* [37] which we outline below in the context of our system. The process involves the following five phases.

Initialization phase: The credential producer chooses a large prime p and α as a primitive root modulo p. In addition, the producer randomly chooses a number x $(2 < x < (p-2))$ and then computes $y = \alpha^x \mod p$. The producer publishes (y, α, p) as the public key, keeps x as the private key, and chooses a one-way hash function $h(g)$ such as SHA-1.

Blinding phase (client and credential producer): The client has a message m and wants to have it signed by the producer. First, the client sends a request to the producer for signing the message m. The producer then randomly chooses a number $\tilde{k}$, such that $\gcd(\tilde{k}, \phi(p)) = 1$. The producer computes $\tilde{r} = \alpha^{\tilde{k}} \mod p$. After computing $\tilde{r}$, the producer sends $\tilde{r}$ to the client. When the client receives $\tilde{r}$ from the producer, the client randomly chooses the set of values (a, b, c), so that parameters (a, b, c) are relatively prime to the value $\phi(p)$. The client then computes $r = \tilde{r}^a y^b \alpha^c \mod p$ and the hash value $h(m)$ generated by the hash function $h(g)$. The client then blinds m with the equation $\tilde{m} = a^{-1}(c + m + r) - \tilde{r} \mod \phi(p)$. After that, the client sends $\tilde{m}$ to the signer.

Signing phase (producer): When receiving the value $\tilde{m}$, the producer computes $\tilde{s} = (\tilde{k} + (\tilde{m} + \tilde{r}))x^{-1} \mod \phi(p)$. The producer then sends the value $\tilde{s}$ to the client.

Unblinding phase (client): After receiving $\tilde{s}$ from the producer, the client computes $s = a\tilde{s}+b \mod \phi(p)$ and obtains the message-signature (m, r, s). The client can then send the message-signature pair (m, r, s) to the credential consumer.

Verifying phase (credential consumer): When the consumer receives the message-signature pair (m, r, s), the verifier can use the one-way hash function $h(g)$ and the public key (y, α, p) to verify the legitimacy of the signature by checking $V_1 = V_2$ so that, $V_1 = y^s \mod p$ and $V_2 = r\alpha^{r+h(m)} \mod p$. If $V_1 = V_2$, then the verification passes; else the verification fails.

4.2 Producing At-Large Credentials

To produce an at-large credential, clients and credential producers play the requester and signer roles, respectively, in the blind signatures signing process (§4.1).

In practice, we assume that each credential consumer has a unique, publicly known identity, analogous to the application identity in the OAuth protocol. For a given consumer, we refer to this identity as id_c. In order to prevent replay attacks in which one credential consumer can use a client's successful login to impersonate the client at another consumer, at-large credentials are bound to a specfic consumer.

Initialization & blinding phases. To produce an at-large credential, a client first chooses a value r to serve as the credential's unique identifier. The value r must be kept secret and should be uniquely chosen for each at-large credential a client obtains. The client binds the identifier to a specific consumer to obtain the message $m = H(r, id_c)$ and requests a blind signature on m from at least t of the n credential producers, uniquely blinding the message m for each of the requests. In this way, the producers learn neither the credential identifier nor the identity of the consumer the credential is for.

Signing & unblinding phases. Upon receiving a signature request, a credential producer must first verify the requesting client's federated identity with the appropriate provider(s) (*e.g.* Facebook and PayPal). If verification succeeds, the producer then checks any rate-limit restrictions for the federated identity. If the limit has not been exceeded, the producer reponds with a blinded signature s'_i on (blinded) message m. The client waits for successful responses from at least t of the n credential producers and unblinds the signatures to obtain vector $s_1, s_2, \ldots, s_t$. This vector serves an at-large credential with identifier r, valid for the credential consumer with identity id_c.

4.3 Consuming At-Large Credentials

Credential consumers can specify which at-large credentials they accept based on the threshold of the credential, dictating the number of signatures required. To authenticate with a credential consumer with identity id_c and requiring a threshold t at-large credential, a client must provide the credential consumer with a credential identifier r along with a vector $s_1, s_2, \ldots, s_t$ of signatures from at least t *unique* credential producers. The consumer first hashes the provided credential identifier with its own identity to reproduce message $m = H(r, id_c)$ and then uses the public keys of the credential producers to verify that each of the provided signatures is, in fact, a valid signature on message m. If at least t signatures verify, the consumer authenticates the client using an anonymous identity derived (using a deterministic, one-way function) from credential identifier r.

4.4 Credential Attributes

Credential attributes allow credential consumers to enforce general restrictions on the at-large credentials they accept. For example, some credential consumers may require that all users be at least 18 years of age. Credential attributes can also be used to provide a higher barrier to entry by requiring, for example, that a Facebook identity has been active for at least one year.

The at-large credential scheme supports credential attributes by using partially blind signatures. *Partially blind signatures* [1] are a modification to blind signatures in which part of the message, the *info* tag, remains visible to the signer. This allows the signer and verifiers to share additional information about the context of the blind signature.

Clients bind attributes to at-large credentials by including each desired attribute in the info tag of their signing requests. Each credential producer then additionally verifies all of the attributes with the federated identity provider and produces a signature only if all check out. Credential consumers enforce credential attributes by ensuring that each signature presented by the client contains all required attributes in the info tag. An inherent restriction on credential attributes is that they must be verifiable with the federated identity provider.

4.5 Rate-Limits via Attributes

Credential attributes additionally allow for finer control over the rate-limits imposed on at-large credential production. Rather than relying on producers to choose a proper default rate for all applications (*e.g.* x credentials per identity per week), clients can instead specify a time interval (*e.g.* 3 days) with each credential request; only if the producer has not already issued the federated identity an at-large credential during the preceding interval does production succeed. As credential attributes are accessible by credential consumers, a consumer can inspect the interval associated with each credential and in turn elect to accept only those credentials satisfying criteria the consumer itself defines. Some consumers may accept 1-hour credentials while other, more abuse-conscious consumers may require 1-month credentials, allowing consumers themselves to determine the degree of accountability they wish to enforce.

4.6 Security/Privacy Properties

The at-large credential scheme is designed to provide the following properties, which correspond to our privacy goals:

- **Anonymity** During credential production, the producers learn the user's federated identity, but not the anonymous identity derived from the credential. During credential consumption, the third-party applications only learn the credential, but not the user's federated identity. Only the user herself knows both pieces needed to complete the mapping between identities.
- **Unlinkability** For any two at-large credentials, neither credential producers nor consumers can determine whether they correspond to the same federated identity.
- **Accountability** Each at-large credential is bound to a unique identifier. If a user misbehaves, the credential with which the user authenticated can be blacklisted by the consumer, effectively banning the user just as with any non-anonymous identity. Producers provide abuse resistance by limiting the rate at which new credentials are assigned to each federated identity.
- **Unforgability** Each at-large credential requires a signature from t of the n credential producers; no colluding group including fewer than t dishonest producers can produce a forged credential that will be accepted by an honest consumer.

We provide a security analysis of how Crypto-Book provides these properties and which attacks it does and does not protect against in our technical report [8].

4.7 Discussions

Crypto-Book's current at-large credential scheme requires the client to obtain a separate blind signature for each third-party application or site the user wishes to visit for the first time, to protect the user from being linked across applications. This limitation may be an inconvenience, especially if Crypto-Book's rate-limits interfere with a user's legitimate attempts to explore several third-party sites in a short time period. Adopting a more sophisticated cryptographic credential scheme such as BLAC [4, 38] might allow the client to pick up a single credential and then "re-blind" it for use across multiple sites while maintaining cryptographic unlinkability and abuse-resistance. Since Crypto-Book's immediate goal is not to find the "ultimate" cryptographic credential scheme but to fit cryptographic credentials (of any kind) into a usable OAuth-compatible architecture, we leave more advanced at-large credential schemes to future work.

5. GROUP CREDENTIAL SCHEME

In this section, we plug another credential scheme, *group credential scheme*, into our Crypto-Book architecture.

As an alternative to at-large credentials, in which anonymity sets form implicitly, group credentials allow an individual to authenticate explicitly as some member of a larger, well-defined set of users. We describe a group credential scheme construction based on linkable ring signatures in this section.

5.1 Building Block: Ring Signatures

Ring signatures [35] rely on group signatures [10] and allow third-parties to verify that a message was signed by one of a well-defined set of private keys, but do not reveal which specific key. Ring signatures are particularly useful for associating properties of a group as a whole, such as credibility in our CB-Drop example, with the signed message. Liu *et al.* propose *linkable ring signatures* (LRS) [26]. LRS is an extended version of ring signatures with the additional property of *linkability* – for any two linkable ring signatures, a third party can determine whether or not the two signatures were produced using the same private key by comparing the *linkage tag* properties of the signatures. We use the linkability property to add accountability to anonymous credentials. The LRS process consists of four phases.

Initialization phase (credential producer and client): Let $G = \langle g \rangle$ be a group of large prime order q. Let $H_1 : \{0,1\}^\star \to \mathbb{Z}_q$ and $H_2 : \{0,1\}^\star \to G$ be independent hash functions. Each member (e.g. Facebook profile) i ($i = 1, ..., n$) has a distinct public key y_i, and private key x_i (assigned by the credential producers) so that $y_i = g^{x_i}$. Let $L = \{y_1, ..., y_n\}$ be the list of n public keys (assigned to federated identities by producers).

Signature generation phase (client): Given the list of public keys L, which the client collects from the producers, the client first uses her private key x_π, which corresponds to her public y_π ($1 \leq \pi \leq n$), to compute $h = H_2(L)$ and $\tilde{y} = h^{x_\pi}$. Then, she picks $u \in \mathbb{Z}_q$ and computes $c_{i+1} = H_1(L, \tilde{y}, m, g^u, h^u)$, where m is the message the client wants to sign with an LRS. Next, for $i = \pi+1, ..., n, 1, ..., \pi-1$, the client picks $s_i \in \mathbb{Z}_q$ and computes $c_{i+1} = H_1(L, \tilde{y}, m, g^{s_i} y_i^{c_i}, h^{s_i} \tilde{y}^{c_i})$. Finally, the client computes $s_\pi = u - x_\pi c_\pi \mod q$. The LRS for the message m is $\sigma_L(m) = (c_1, s_1, ..., s_n, \tilde{y})$.

Verification phase (credential consumer): Suppose a credential consumer receives a message m signed by $\sigma_L(m) = (c_1, s_1, ..., s_n, \tilde{y})$ and the consumer knows the public key list L (obtained from the producers), the consumer first computes $h = H_2(L)$. For $i = 1, ..., n$, the consumer computes $z_i' = g^{s_i} y_i^{c_i}$ and $z_i'' = h^{s_i} \tilde{y}^{c_i}$ and then $c_{i+1} = H_1(L, \tilde{y}, m, z_i', z_i'')$ if $i \neq n$. Finally, the consumer checks if $c_1 = H_1(L, \tilde{y}, m, z_n', z_n'')$. If so, the consumer accepts the signature and the client is authenticated; otherwise authentication fails.

Linkability checking (credential consumer): Given two signatures $\sigma_L'(m') = (c_1', s_1', ..., s_n', \tilde{y}')$ and $\sigma_L''(m'') = (c_1'', s_1'', ..., s_n'', \tilde{y}'')$ corresponding to messages m' and m'', the consumer can check whether the two signatures are from the same signer by checking if $\tilde{y}' = \tilde{y}''$. This allows the consumer to maintain a consistent pseudonym for each client.

5.2 Producing Group Credentials

A group credential consists of two components - the user's individual private key and the set of public keys of all other members of the desired anonymity set.

Credential production process begins with an initialization phase. Suppose a user Alice wants to collect her private key. She first connects to a credential producer through an anonymity network (such as Tor) and supplies a list of federated identities, including her own, as her desired anonymity set. The credential producers then work together to generate public/private key pairs for each federated identity using Pederson's PKG [35], in which all credential producers contribute to the overall value of the credential but each learns only a single share. Finally, for each identity in the anonymity set, each of the n credential producers holds a single share of the corresponding key, t of which are necessary to reconstruct the key.

After all keys have been generated, each credential producer sends an invitation to each of the federated identities (via Facebook message, for example) inviting them to join the Crypto-Book service and collect their private key. This indirection is necessary as if a user directly requested their private key a credential producer colluding with a credential consumer could potentially de-anonymize a user via a timing analysis attack by correlating the private key request with subsequent authentications.

Alice (and, independently, the other identities who have received invitations) then follows the invitations to collect a share of her private key from each of the credential producers. Before releasing the share, each producer first requires Alice to authenticate with her federated identity provider via OAuth, proving that she in fact owns the identity corresponding to the private key. After collecting shares from at least t of the n credential producers, Alice combines the shares to recover her private key.

Obtaining the set of public keys requires no interaction between client and federated identity provider. Instead, Alice directly contacts each credential producer and requests a share of the public key for each identity in her anonymity set. After receiving at least t shares of each key, Alice recovers the set of public keys. In practice, clients bundle all key requests for a given anonymity set and producers return all shares in a single response, to minimize transmission overhead and latency.

5.3 Consuming Group Credentials

Credential consumers authenticate group credentials by requiring users to supply a valid linkable ring signature over a message of the consumer's choosing; typically, a fresh, random value. Such a challenge prevents replay attacks and ensures that the user knows a valid private key for the ring.

The user first contacts the credential consumer with an authentication request. The consumer then replies with a challenge, which the user signs using their group credential and returns to the consumer. In some instances, the user also supplies the anonymity set to which the group credential corresponds; in others, this set is implicitly determined by the consumer itself. In either case, the consumer first asserts that the anonymity set contains valid Crypto-Book identities for the given consumer.

The consumer then collects public key shares for all members of the anonymity set from at least t credential producers and verifies the ring signature against the resulting public keys. If the signature verifies, authentication succeeds and the consumer maps the user to an anonymous account based on the linkage tag of the signature, creating a new account if this was the first authentication for the given tag.

5.4 Security/Privacy Properties

The group credential scheme is designed to provide the following properties:

- **Anonymity** During credential production, each credential producer learns only a single share of a user's private key. No colluding group of fewer than t dishonest produces can recover the private key needed to de-anonymize the user.
- **Accountability** Linkability of the signatures produced by group credentials ensures that a given credential always maps to the same anonymous identity. Thus group credential identities can be banned just as any non-anonymous identity.
- **Unforgability** Any valid group credential must include a private key for a valid Crypto-Book identity. Shares for such a key must be obtained from at least t credential producers. Thus no colluding group including fewer than t dishonest producers can produce a forged credential that will be accepted by an honest consumer.

We provide a security analysis of how Crypto-Book provides these properties and which attacks it does and does not protect against in our technical report [8].

5.5 Discussions

Crypto-Book's current group credential scheme uses linkable ring signatures whose size is linear in the anonymity set size; their efficiency on large anonymity sets could be improved using accumulator-based schemes [6, 7] at the cost of more complex computations.

Another disadvantage is that using any form of signatures for authentication leaves a non-repudiable trail, which might expose a user whose private signing key is later compromised. This limitation might be addressed by adopting techniques from deniable authentication protocols [15, 31].

Finally, in practice it may be hard for users to pick "good" anonymity sets. If all the other users a whistleblower conscripts into his "anonymity set" turn out to be implausible for some reason to an investigating adversary, *e.g.*, because none of the other members could have had access to the leaked document, then the chosen anonymity set may prove ineffective. We make no suggestion that group credentials are straightforward to use safely: only that, if the user's only other alternative is to disclose his identity completely (*e.g.*, to persuade the journalist of his credibility), then group anonymity may be better (and perhaps at least more "plausibly deniable") than no anonymity.

6. IMPLEMENTATION

We have implemented a Crypto-Book prototype. The prototype interfaces with Facebook, PayPal, or a combination of the two, as federated identity providers. We implemented credential producers, consumers, and clients for both at-large and group credentials and deployed the system on a distributed set of servers. Users can collect credentials from these producers using their existing federated identities.

We implemented at-large credentials based on blind [37] and partially blind signatures [1] and developed standalone credential producers and consumers for both schemes. For group credentials we implemented RSA-based ring signatures [35] and DSA-based linkable ring signatures [26]. We implemented (t, n)-threshold DSA key pair generation for group credentials using Pederson's distributed PKG [35].

In order to maintain the convenience users have come to expect of federated authentication, we have implemented a web-based Crypto-Book client, named CB-Login. CB-Login combines an application-embedded consumer with a Google Chrome extension to offer a one-click login experience akin to a privacy-preserving "Login with Facebook". The Chrome extension retrieves and manages group credentials and generates linkable signatures in the web browser, uploading the signatures to the CB-Login consumer. The consumer verifies the signatures and produces anonymous identities based on the linkage tags of the signatures. CB-Login requires no more user interaction than a traditional federated authentication.

6.1 Applications

We have implemented three applications on top of the Crypto-Book architecture: CB-Wiki, CB-Dissent and CB-Drop, concretizing the motivating use-cases outlined in §2.2. In addition, we also implemented OAuth provider functionality in our client CB-Login to allow other applications to more easily incorporate Crypto-Book as a federated identity provider.

CB-Wiki is a Wiki system based on MediaWiki [30] that allows users to log in as *anonymous* yet *accountable* pseudonymous identities rather than *personally identifiable* users. This design benefits both users, who can edit Wiki pages without disclosing their identity, and administrators, as if a user repeatedly vandalizes a page or conducts other forms of system abuses an administrator can issue warnings or other sanctions (*i.e.* banning) just as for any traditional, non-anonymous user.

CB-Dissent is a system that combines the Crypto-Book anonymous authentication architecture with the Dissent [12, 41] scalable anonymous messaging system. CB-Dissent is an anonymous authentication system that allows users to anonymously request a Dissent session (a set of Dissent servers) to be started and then to anonymously authenticate themselves and connect to that Dissent group. Unlike in traditional Dissent, in CB-Dissent servers do not learn the actual identities of the clients connected to them, but only that the client is a valid Crypto-Book identity.

CB-Drop builds on SecureDrop [36], an open-source whistleblower submission system developed by the Freedom of the Press Foundation. SecureDrop allows journalists to accept sensitive documents from anonymous sources via a web interface running as a Tor hidden service. CB-Drop adds credibility to leaks by allowing a source to anonymously sign a document using a relevant set of Crypto-Book identities before submitting it via SecureDrop. Upon retrieving the document, a journalist can then verify the signature, increasing confidence in the authenticity of the leak without compromising the source's anonymity.

7. EVALUATION

To evaluate the practicality of Crypto-Book we consider both end-to-end measurements in expected deployment scenarios and microbenchmarks focusing on scalability of specific components. We first describe the experimental setup and then evaluate credential production and consumption in both proposed credential schemes. We conclude by evaluating Crypto-Book in the context of our example applications. Evaluations of CB-Drop, as well as code modification metrics for CB-Wiki, CB-Dissent and CB-Drop are available in our technical report [8].

7.1 Experimental Setup

The experimental setup for the following evaluations include three classes of machines, based on role in the system:

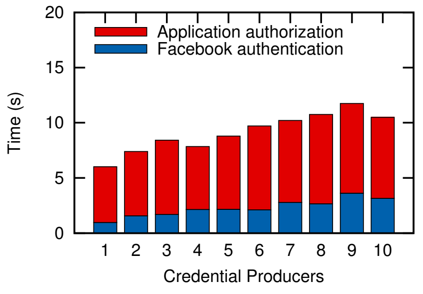

Figure 3: Facebook application authorization

Key Parameters	Signature Size (Bytes)
(1024,160)	210
(2048,224)	287
(2048,256)	325
(3072,256)	326

Table 1: Partially blind signature size

- **Clients** are consumer laptops with 2.4GHz Intel Core i5 processors and 8GB of RAM.
- **Credential producers** are nodes on the geographically distributed PlanetLab [11] network. A typical PlanetLab node has a 2.4GHz Intel Xeon processor and 4GB of RAM.
- **Credential consumers** are commercial shared hosting providers also with 2.4GHz Intel Xeon processors and 16GB of RAM.

In selecting experimental key pairs we followed NIST recommendations [33] for DSA keys where the tuple (L, N) specifies the bit-length of the p and q parameters, respectively. If a parameter tuple is not specified, it is assumed to be $(1024, 160)$.

7.2 Producing Credentials

In this section we evaluate the production of credentials in both the at-large and group credential schemes.

7.2.1 Facebook Application Authorization

In both the at-large and group credential schemes a first-time user must authorize a credential producer's application with a supported federated identity provider before retrieving any credentials from that producer. We evaluated the time taken to complete this step for a varying number of credential producers, with Facebook as the federated indentity provider. We used our Chrome extension to automate the authorization process and performed all authorizations in parallel. Results are presented in Figure 3, drawing distinction between time spent authenticating with Facebook and time spent authorizing the Crypto-Book application. The times shown are those of the last credential producer to return. **A client only ever needs to perform this setup step once, the first time they ever use Crypto-Book.**

7.2.2 At-Large Credentials

An at-large credential consists of partially blind signatures from t credential producers. As each signature is obtained independently from all others, we focus on time taken to obtain a single signature. As discussed in Section 4, a signature is generated collaboratively by the client and a producer. Network overhead consists of two round trips between producer and client where the second trip is dependent on the size (and hence strength) of the signature. Signature sizes for varying signing key sizes are shown in Table 1.

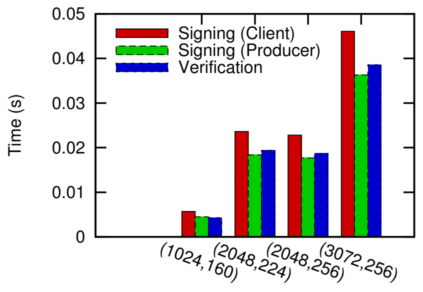

Figure 4: Partially blind signature operations

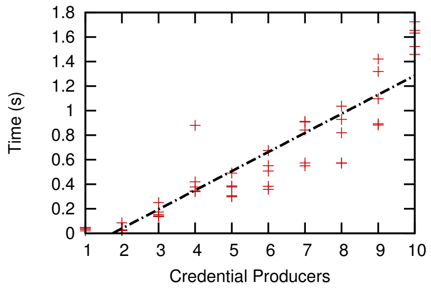

Figure 5: Distributed keypair generation

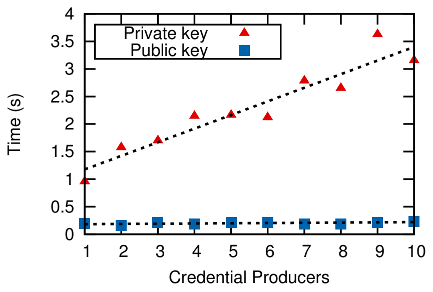

Figure 6: Retrieval of previously generated keys

Entity	Operation	Time (s)
Client	Produce LRS	0.257
Credential Consumer	Fetch Public Keys	1.011
	Verify LRS	0.035
Client-Consumer Network Latencies		0.304
Total User-Observable		**1.607**

Table 2: End-to-end Group Authentication

The computation costs for partially blind signature operations are shown in Figure 4. Credential production consists only of the signing operation, which, for a 2048-bit signing key, takes approximately 50ms of computation time, with effort divided fairly evenly between client and producer.

7.2.3 *Group Credentials*

Keypair Generation In our group credential scheme, the first time a key pair is requested it must be collectively generated by the credential producers. We evaluated the time required to generate a single key pair for a varying number of credential producers and present the results in Figure 5. Times shown include only operations involving producers; returning the keys to the client is evaluated separately. For a reasonable number of producers, we find that the results scale approximately linearly.

Key Retrieval We next evaluate the time taken for a client to retrieve a single key, either private or public, from the set of credential producers. Requests to all producers are performed in parallel. We present the results in Figure 6. For public keys, we find near constant response. Times shown for private keys include Facebook authentication, which accounts for the difference when compared to public key requests. **Private key requests will be rare, as after retrieval users retain their private keys locally, stored in the Chrome extension.**

7.3 Consuming Credentials

In this section we evaluate the consumption of credentials in both the at-large and group credential schemes.

7.3.1 *At-Large Credentials*

In evaluating the consumption of at-large credentials we assume that all credential producers' signing keys are well-known to all credential consumers. As a result, authentication via at-large credentials consists only of the transmission and verification of partially blind signatures. As shown previously in Figure 1, a threshold t at-large credential is approximately $300t$ bytes in size; at these sizes credential upload times are heavily dependant on network properties.

To evaluate the costs of verification we considered $t = 1$ at-large credentials for varying key size. Results are shown in Figure 4; for a 2048-bit key verification takes less than 20ms. As each signature verification is completely independent of all others, for expected values of t (≤ 10), signature verification can be performed largely in parallel.

7.3.2 *Group Credentials*

In order to perform an **end-to-end evaluation** of authentication using group credentials, we first created a group credential including ten Facebook identities belonging to members of the authors' research group. Users then each collected their group credential and used our Chrome extension to authenticate to an internal website with their Crypto-Book identity. We used an application-embedded consumer and three credential producers on networks different from the consumer's. We recorded timings for each phase of 117 authentications and present the averages in Table 2. On average, the total user-observable authentication time was 1.6 seconds; this is approximately a 1.2 second overhead compared to non-anonymous federated authentication with Facebook.

To investigate how group credential authentication scales with group size, we considered each operation from Table 2 separately. We found that by bundling public key requests we were able to fetch up to 1000 public keys in near-constant time, identifying the ring signature operations as the limiting factor for larger groups. We then measured the computation time for each ring signature operation, varying the ring size between 1 and 1000. Results for signing and verification are shown in Figures 7 and 8, respectively. Both operations scale linearly with ring size and, for ring sizes near 100 and 2048-bit keys, both operations complete in one second.

We additionally measured the size of the linkable ring signature produced for varying ring sizes. This signature is what the client sends to the credential consumer with each authentication, thus overall client-consumer network latency depends on signature size. Results are shown in Figure 9. We found that signature size scales linearly with ring size and that, for ring sizes near 100 and 2048-bit keys, signatures are less than 10kB.

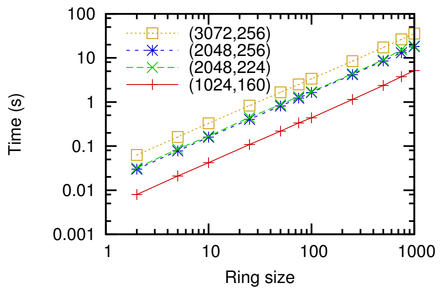

Figure 7: Linkable ring signature generation

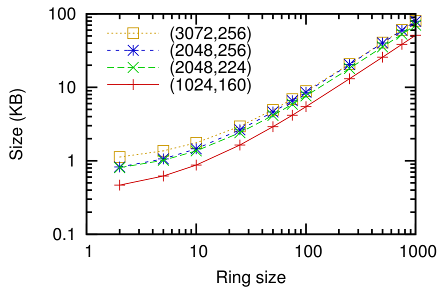

Figure 9: Linkable ring signature size

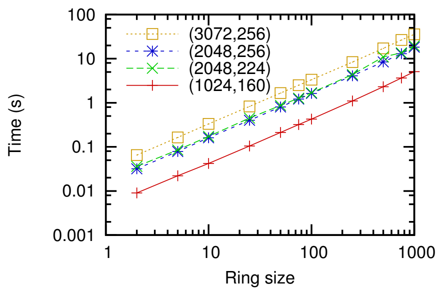

Figure 8: Linkable ring signature verification

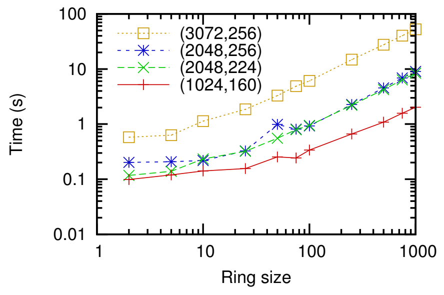

Figure 10: CB-Dissent authentication time

7.4 CB-Dissent Authentication

Finally, we evaluated group authentication in our CB-Dissent implementation. This experiment measures the time required for a client to authenticate with a single Dissent server acting as an application-embedded consumer. We again varied the group size between 1 and 1000, presenting the results in Figure 10. For a group size of 100 and 2048-bit keys, authentication again takes under 1 second. While this is a significant overhead compared to traditional pre-exchanged key authentication, it remains well within practical limits.

8. RELATED WORK

Existing work on anonymous credential systems include BLAC [4, 38] which supports blacklisting of anonymous credentials in certain situations. There have been a wide variety of other approaches to anonymous credential systems including those based on group signatures [2, 3, 5, 10], dynamic accumulators [6, 7] and Nymble systems [22, 23].

Felt and Evans [18] examine privacy protection in social networking APIs. The deployment of public key cryptography over social networks was considered by Narayanan *et al.* [32] where they considered key exchange over social networks. They considered using social networks as a public key infrastructure (PKI), they did not implement any applications that use the public keys.

Various schemes have been proposed to protect user data *within* an online social network [13, 19, 27, 28], by encrypting the content stored within the social network. However these schemes did not consider the privacy risks involved when a user uses their online social networking identity to identify themselves with third parties such as logging into other websites using their Facebook credentials. Dey and Weis [14] proposed PseudoID, a similar system based on blind signatures for privacy protected federated login, however their scheme does not handle key assignment as our work does. A similar blind signature based system was proposed by Khattak *et al.* [24]. Watanabe and Miyake [39] made initial efforts towards account checking however still did not consider key assignment. Opaak [29] is a system that attempts to provide some Sybil resistance through relying on a cellphone as a scare resource. SudoWeb [25] looked at limiting the amount of Facebook information disclosed to third party sites but did not consider fully anonymous online IDs.

9. CONCLUSIONS

We have demonstrated Crypto-Book, a practical and pluggable architecture for providing privacy preserving online identities based on federated identity providers. As two examples, this paper deploys two different credential schemes into Crypto-Book through adapting two cryptographic primitives, blind signature and linkable ring signature, respectively. The two deployments are capable of supporting different needs of privacy protection.

We have implemented three major applications, CB-Wiki, CB-Dissent, and CB-Drop, on top of Crypto-Book and shown them to have good scalability properties. We believe that Crypto-Book is a practical way to provide federated identity users with accountable pseudonyms and many applications could be developed on top of Crypto-Book.

There remain a large number of areas for future research based on our architecture as well as many applications that could be developed on top of Crypto-Book leaving open a wide range of areas for investigation building on our results in future work.

Acknowledgments

We wish to thank the anonymous CODASPY reviewers as well as Joan Feigenbaum, Ramki Gummadi and Anil Somayaji for their valuable feedback. This work is supported in part by NSF grants CNS-1407454 and CNS-1409599.

10. REFERENCES

[1] Masayuki Abe and Tatsuaki Okamoto. Provably secure partially blind signatures. In *20th CRYPTO*, 2000.

[2] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *20th CRYPTO*, 2000.

[3] Giuseppe Ateniese, Dawn Song, and Gene Tsudik. Quasi-efficient revocation of group signatures. In *7th FC*, January 2003.

[4] Man H Au, Apu Kapadia, and Willy Susilo. BLACR: TTP-free blacklistable anonymous credentials with reputation. In *19th NDSS*, February 2012.

[5] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *24th CRYPTO*, 2004.

[6] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *22nd CRYPTO*, August 2001.

[7] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT*, May 2001.

[8] Technical Report: Providing Abuse Resistant Pseudonyms for Federated Online Identities with Cobra. https://www.dropbox.com/s/c90jn2om7ahlth4/anon-tech-report.pdf?dl=0.

[9] David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, 1982.

[10] David Chaum and Eugène Van Heyst. Group signatures. In *EUROCRYPT*, April 1991.

[11] Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. Planetlab: an overlay testbed for broad-coverage services. *ACM SIGCOMM Computer Communication Review*, 33(3):3–12, 2003.

[12] Henry Corrigan-Gibbs and Bryan Ford. Dissent: accountable anonymous group messaging. In *17th CCS*, October 2010.

[13] Leucio Antonio Cutillo, Refik Molva, and Thorsten Strufe. Safebook: A privacy-preserving online social network leveraging on real-life trust. *Communications Magazine, IEEE*, 47(12):94–101, 2009.

[14] Arkajit Dey and Stephen Weis. Pseudoid: Enhancing privacy in federated login. *Hot topics in privacy enhancing technologies*, pages 95–107, 2010.

[15] Mario Di Raimondo, Rosario Gennaro, and Hugo Krawczyk. Deniable authentication and key exchange. In *CCS*, 2006.

[16] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: the second-generation onion router. In *13th USENIX Security Symposium*, August 2004.

[17] John R. Douceur. The Sybil attack. In *1st IPTPS*, March 2002.

[18] Adrienne Felt and David Evans. Privacy protection for social networking APIs. *W2SP*, 2008.

[19] Saikat Guha, Kevin Tang, and Paul Francis. Noyb: Privacy in online social networks. In *WOSN*, 2008.

[20] E. Hammer-Lahav. The OAuth 1.0 protocol, April 2010. RFC 5849.

[21] Ed Hardt. The OAuth 2.0 authorization framework, October 2012. RFC 6749.

[22] Ryan Henry, Kevin Henry, and Ian Goldberg. Making a nymbler nymble using verbs. In *PETS*, 2010.

[23] Peter C Johnson, Apu Kapadia, Patrick P Tsang, and Sean W Smith. Nymble: Anonymous IP-address blocking. In *PETS*, 2007.

[24] Zubair Ahmad Khattak, Jamalul-lail Ab Manan, Suziah Sulaiman, et al. Analysis of open environment sign-in schemes-privacy enhanced & trustworthy approach. *Journal of Advances in Information Technology*, 2(2):109–121, 2011.

[25] Georgios Kontaxis, Michalis Polychronakis, and Evangelos P Markatos. SudoWeb: Minimizing information disclosure to third parties in single sign-on platforms. In *Information Security*, pages 197–212. Springer, 2011.

[26] Joseph K Liu and Duncan S Wong. Linkable ring signatures: Security models and new schemes. In *ICCSA*, May 2005.

[27] Matthew M Lucas and Nikita Borisov. Flybynight: mitigating the privacy risks of social networking. In *Proceedings of the 7th ACM workshop on Privacy in the electronic society*, pages 1–8. ACM, 2008.

[28] Wanying Luo, Qi Xie, and Urs Hengartner. Facecloak: An architecture for user privacy on social networking sites. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 3, pages 26–33. IEEE, 2009.

[29] Gabriel Maganis, Elaine Shi, Hao Chen, and Dawn Song. Opaak: using mobile phones to limit anonymous identities online. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pages 295–308. ACM, 2012.

[30] MediaWiki. http://www.mediawiki.org.

[31] Moni Naor. Deniable ring authentication. In *22nd CRYPTO*, August 2002.

[32] Arvind Narayanan, Narendran Thiagarajan, Mugdha Lakhani, Michael Hamburg, and Dan Boneh. Location privacy via private proximity testing. In *Proc. of NDSS*, volume 2011, 2011.

[33] The FIPS 186-4 Digital Signature Algorithm Validation System. http://csrc.nist.gov/groups/STM/cavp/documents/dss2/dsa2vs.pdf.

[34] David Recordon and Drummond Reed. OpenID 2.0: A platform for user-centric identity management. In *Proceedings of the second ACM workshop on Digital identity management*. ACM, 2006.

[35] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *7th ASIACRYPT*, December 2001.

[36] SecureDrop. https://pressfreedomfoundation.org/securedrop/.

[37] Victor R. L. Shen, Yu fang Chung, Tzer Shyong Chen, and Yu An Lin. A blind signature based on discrete logarithm problem. *ICIC*, 7(9), September 2011.

[38] Patrick P Tsang, Man Ho Au, Apu Kapadia, and Sean W Smith. Blacklistable anonymous credentials: Blocking misbehaving users without TTPs. In *14th CCS*, October 2007.

[39] Ryu Watanabe and Yutaka Miyake. Account management method with blind signature scheme. *Engineering and Technology, World of Science*, (59):2069–2073, 2011.

[40] Alma Whitten and J. Doug Tygar. Why johnny can't encrypt: A usability evaluation of PGP 5.0. In *Proceedings of the 8th USENIX Security Symposium*, August 1999.

[41] David Isaac Wolinsky, Henry Corrigan-Gibbs, Aaron Johnson, and Bryan Ford. Dissent in numbers: Making strong anonymity scale. In *10th OSDI*, October 2012.