ANONIZE: A Large-Scale Anonymous Survey System

Susan Hohenberger
Johns Hopkins University
susan@cs.jhu.edu

Steven Myers
Indiana University
samyers@indiana.edu

Rafael Pass
Cornell University
rafael@cs.cornell.edu

abhi shelat
University of Virginia
abhi@virginia.edu

July 6, 2015

Abstract

A secure *ad-hoc survey* scheme enables a survey authority to independently (without any interaction) select an ad-hoc group of registered users based only on their identities (e.g., their email addresses), and create a survey where only selected users can *anonymously* submit *exactly one* response. We present a formalization of secure ad-hoc surveys and present:

- an abstract provably-secure implementation based on standard cryptographic building blocks (which in particular are implied by the existence of enhanced trapdoor permutations in the CRS model);
- a *practical* instantiation of our abstract protocol, called ANONIZE, which is provably-secure in the random oracle model based on cryptographic assumptions on groups with bilinear maps.

As far as we know, ANONIZE constitutes the first implementation of a large-scale secure computation protocol (of non-trivial functionalities) that can scale to millions of users.

1 Introduction

We study the basic conflict between anonymity and authenticity in large network settings. Companies, universities, health providers and government agencies routinely conduct asynchronous and real-time data collection surveys for targeted groups of users over the Internet. To do so, they aim for *authenticity* (i.e., ensuring that only the legitimate users can participate in the data collections) and *anonymity* (i.e., ensuring that the there is no link between the legitimate user and his/her data so that users are more likely to submit honest feedback). The intrinsic conflict between these two goals may result in users self-censoring or purposely biasing data they submit.

A simple example is a course evaluation for a university class. A typical implementation of such a survey requires a trusted third party (such as the university or some external party) to ensure that feedback is collected anonymously from the participants and that only authorized participants, i.e., the students enrolled in a particular class, can submit feedback for that class. In such trusted-party implementations, students are required to authenticate themselves with their university IDs and thus leave a link between their evaluation and their identity; they are trusting the survey collector to keep such links private.

Assuming that the survey collector acts as a trusted third party is dangerous. Even if the survey collector intends to keep the links between users and their surveys private, its computer may be stolen or broken into, and the information leaked. For instance, in 2009, a computer at Cornell was stolen, containing sensitive personal information, such as name and social security number, for over 45,000 current and former university members [1]. Additionally, even if users have full confidence in the trusted third party, and in particular, its ability to keep its data *secure*, developing an anonymous survey system using such a trusted party still requires extreme care. For example, in the implementation of course reviews at the University of Virginia, side channel information about participation in the survey may leak information about the order in which students participate. Later, the order of the students' comments in the aggregated responses may be correlated to break anonymity [2].

Furthermore, in many situations, jurisdictional boundaries or legal requirements make it unfeasible to rely on solutions with external trusted third parties: it may be illegal to store sensitive patient information on a third-party system; similarly, many countries do not permit sensitive data to be stored on servers run by foreign corporations due to the potential for this data to be seized [3].

For these reasons, we seek cryptographic solutions to the problem of anonymous surveys that offer security guarantees where anonymity and authenticity hold *without needing to trust a third party*.

Cryptographic voting techniques described in prior work may offer a partial solution to this problem (see e.g., [4, 5, 6, 7, 8]). In such schemes, each survey consists of two steps: 1) users authenticate themselves to a server and anonymously check out a

single-use "token"; the token itself carries no link to the user's identity. 2) a user can then use her token to participate in the specified survey. Such schemes provide good anonymity *assuming that* users actually separate steps 1 and 2 with a reasonably long time lag (otherwise there is a clear time link between the user and its data). But if users are required to separate the two steps by, say, a day, the ease-of-use of the survey is significantly hampered and become much less convenient than "non-anonymous" surveys (or anonymous surveys employing a trusted third party). Additionally, the extra steps required to authenticate for each survey may be onerous. Consequently, such techniques have gained little traction.

1.1 Our innovation: electronic ad-hoc surveys

In this paper, we consider a general solution to the problem of anonymously collecting feedback from an authenticated group of individuals by introducing the notion of an *ad-hoc survey*. The "ad-hoc" aspect of this notion means that *anyone* can select a group of individuals and create a survey in which those and only those individuals can complete the survey *at most once*; additionally, the survey initiator can initiate this survey knowing only the identities (e.g., the email addresses) of the users in the ad-hoc group—no further interaction between the survey initiator and the users is required.[1] As such, our method provides essentially the same ease-of-use as traditional (non-anonymous) electronic surveys (and it thus is expected to increase user participation and make the feedback submitted more valuable).

As we demonstrate, ad-hoc surveys admit practical and efficient solutions for very large surveys: we present an ad-hoc survey scheme, ANONIZE, a proof of security for the cryptographic protocols in ANONIZE, and an implementation of the protocol. ANONIZE supports millions of "write-in" (i.e., collection of arbitrary strings of data) surveys in minutes. As far as we know, this is the first implementation of a provably-secure[2] multi-party protocol that scales to handle millions of users. Additionally, we prove security of our scheme even if the adversary participates in an arbitrary number of concurrent surveys.

1.2 Ad-hoc Surveys in more detail

In more details, there are three parties in an ad-hoc survey system: a registration authority (RA) that issues master user tokens, a survey authority (SA) that can create surveys, and users that provide survey data. A user must first register with the RA and retrieve a secret "master user token". This is a single token that can be used for all future surveys the user participates in. Anyone can act as an SA by choosing a uniquel survey ID and

[1] Before users can complete a survey, we additionally require them to register their identity. We emphasize that this registration is done only once and can be used for any number of subsequent surveys.

[2] By "provably-secure", we only refer to the cryptographic protocol.

publishing a list of identities that are permitted to participate in that survey. The list of identities that can participate in a particular survey can grow dynamically, and the SA can create a survey without any interaction with others. Finally, a user who is on the list of valid identities for a survey can *non-interactively* submit a response to the survey by simply routing one message to the SA (through an anonymous network like Tor, or anonymous proxy relay).

To exemplify this approach and informally discuss the anonymity/authenticity properties it provides, we consider the course evaluation scenario.

1.2.1 Student Registration

When a student is asked to set-up his college/university account information (while proving his identity using traditional, non-electronic, methods), the student also generates an *unlinkable master user token* that is tied to his school email identity (e.g., his email address). This step can also be done at a later stage if the student desires (or if the student loses his credential), but it only needs to be done *once*.

1.2.2 Course Survey Setup

Whenever a course administrator wishes to set-up a course survey, she generates a *survey key* based only on the actual identities (e.g., the email addresses) of the course participants.

1.2.3 Survey Execution

Upon filling out a survey with its associated survey key, the student's client (either computer or smart phone) combines the survey key and her *master user token* to generate an unlikable *one-time token* that she can use to complete the survey. The one-time token satisfies two properties: 1) it carries no link to the student's identity (thus we have anonymity), and 2) for a given survey key, the student can obtain at most one such token (and thus we ensure that a student can only complete the survey once[3]). The results of the survey can now be tabulated, and, possibly announced.

We emphasize that once Step 1 has been done (presumably once the students enroll into college), Steps 2 and 3 can be repeatedly performed. The participants do not need to check-out new single-use tokens for each survey; rather their client uses the master user token to create a unique single-use token for this survey *without any interaction* (that could deanonymize the student).

[3]Our systems support the (optional) ability for the user to change her response (before the voting deadline) in a manner that replaces her previous submission, but in no other way leaks any information about her identity.

Part of our contribution is to precisely define security properties of ad-hoc surveys such as anonymity (intuitively, that there is no link between users and the surveys they submit), and authenticity (intuitively, that only authorized users can complete the survey, and they can complete it only once). As mentioned, we are interested in providing security not only for a single survey, but also if an attacker participates in many surveys, be they in the past, concurrent, or in the future. We emphasize that although related notions of anonymity and authenticity have been defined in the literature for other applications, our setting is considerably more complex and thus the actual definitions are quite different.

1.3 Anonize in more detail

Our system is constructed in two steps. We first provide an *abstract* implementation of secure ad-hoc surveys from generic primitives, such as commitment schemes, signatures schemes, pseudo-random functions (PRF) and generic non-interactive zero-knowledge (NIZK) arguments for NP[4]. We prove the security of the abstract scheme based on the assumption that all generic primitives employed are secure. Note that we have taken explicit care to show that our schemes remain secure even when the adversary initiates *many concurrently* executing sessions with the system.

In a second step we show that (somewhat surprisingly) the generic scheme can be instantiated with *specific* commitment schemes, signatures schemes, PRFs and NIZKs to obtain our efficient secure ad-hoc survey scheme ANONIZE (which now is based on specific computational assumptions related to the security of the underlying primitives in the Random Oracle Model). The surprising aspect of this second step is that our generic protocol does *not* rely on the underlying primitives in a black-box way; rather, the NIZK is used to prove complex statements which require code of the actual commitments, signatures and PRFs used. In this second step, we rely on ideas similar to those underlying efficient constructions of anonymous credentials in bilinear groups [9, 10], although our constructions differ in a few ways. As far as we know, our scheme is also one of the first implementations of a cryptographic scheme that is concurrently secure.

Let us briefly provide a high-level overview which omits several important features, but conveys the intuition of our abstract protocol (we assume basic familiarity with the concepts of commitment schemes, signature schemes, PRFs and NIZKs).

Registration A user with identity *id* registers with the RA by sending a commitment to a random seed s_{id} of a pseudo-random function (PRF) F. If the user has not previously been registered, the RA signs the user's name along with the commitment. The signature σ is returned to the user is its "master user token".

Survey Creation To create a survey with identity vid, an SA generates a signature key pair $(\mathsf{vk}_{\mathsf{SA}}, \mathsf{sk}_{\mathsf{SA}})$ and publishes:

[4] As we show, we actually need a new variant of standard NIZKs.

- the signature verification key $\mathsf{vk}_{\mathsf{SA}}$, and
- a list of signed pairs $\{(\mathsf{vid}, id)\}_{id \in I}$ where I is the set of users authorized for the survey.

Submitting a Response To complete a survey with survey identity vid, user id generates a *single-use token* $\mathsf{tok} = F_{s_{id}}(\mathsf{vid})$ (by evaluating the PRF on the seed s_{id} with input vid) and presents a NIZK that

- it "knows a signature by the RA on *some* identity id and a commitment to some seed s_{id}" (note that neither id nor s_{id} is revealed);
- it "knows a signature by the SA on the pair (vid, id)" (again, id is not revealed); and,
- the single-use token tok is computed as $F_{s_{id}}(\mathsf{vid})$.

The user's actual survey data as well as tok and vid are included into the "tag" of the NIZK to ensure "non-malleability" of submitted responses.

1.3.1 Proof of Security and the Concrete Instantation

Roughly speaking, the NIZK proof in the survey completion step ensures that only authorized users can complete the survey, and that they can compute at most one single-use token, and thus complete it at most once. [5] Anonymity, on the other hand, roughly speaking follows from the fact that neither the RA nor the SA ever get to see the seed s_{id} (they only see commitments to it), the zero-knowledge property of the NIZKs, and the pseudo-randomness property of the PRF.

Proving this abstract protocol secure is non-trivial. In fact, to guarantee security under concurrent executions, we introduce and rely on a new notion of a *online simulation-extractable NIZK* (related to simulation-sound NIZK [11] and simulation-extractable interactive zero-knowledge arguments [12, 13]).

To enable the concrete instantiation of the abstract protocol using specific primitives, we demonstrate a simple and efficient way of implementing online simulation-extractable NIZK in the Random Oracle Model. Finally, the key to the construction is choosing appropriate commitments, signatures and PRF that can be "stitched together" so that we can provide an efficient NIZK for the rather complex statement used in the abstract protocol.

[5] If the user wants to replace her survey response before the deadline and this is allowed by the system, then she can create a new NIZK with new data for the same $F_{s_{id}}(v_{id})$ value. The old survey with this value can be deleted.

1.4 Related notions and techniques

Ad-hoc surveys are related to, but different from, a number of primitives previously considered in the literature such as *group signatures*, *ring signatures*, *voting schemes* and *anonymous credentials*. Roughly speaking, group [14, 15, 16] and ring [17] signatures allow members of a set of users to sign messages in a way that makes it indistinguishable who in the set signed the message (in the case of group signatures the set is fixed [6], whereas in the case of ring signatures the set can be selected "ad-hoc"). This property is similar to the anonymity property of ad-hoc survey, but unfortunately, the authentication property these notions provide is insufficient for our setting—in a ring signature scheme, a user may sign multiple messages with impunity which corresponds to the ability to complete the survey multiple times in our context. Voting schemes [4, 5, 6, 7, 8] on the other hand do provide both the anonymity and the authenticity properties we require; however, they do not allow for the authenticated users to be selected ad-hoc for multiple elections.

An anonymous credential system [19, 20, 21, 9] allows users to obtain credentials from authorities and to anonymously demonstrate possession of these credentials. In essence such systems provide methods for providing, a "zero-knowledge proof of knowledge of a signature on a set of attributes." As mentioned, the NIZKs we use rely on intuitions similar to those used in constructions of anonymous credentials (most closely related to [9] and the electronic cash/token extensions in [22, 10]), but we have different goals and rely on different complexity assumptions. Moreover, since anonymous credentials typically are not analyzed under concurrent executions, we must develop new techniques for the security analysis.

1.5 Our Implementation

One of the key points of our system is that it can be implemented and can easily handle large numbers of users with moderate resources. The computational costs on the users are quite low as well, with a typical desktop being able to compute the worst-case scenario in under a few seconds, using a single core of the machine. Thus we argue our system scales to manage that vast majority of practical surveying needs at costs that are easily affordable.

1.6 Outline of the Paper

We provide some preliminaries in Section 2. In Section 3 we define correctness and security properties of Ad-hoc Surveys. In Section 4 we provide an abstract implementation of Ad-hoc surveys based on general cryptographic building blocks. Finally, in Section 5

[6] The desirability of making group signatures dynamic was addressed by Bellare, Shi and Zhang [18]. Their solution, however, requires that every group member or potential group member has their own personal public key, established and certified, e.g., by a PKI, independently of any group authority. Our ad-hoc survey solution does not require this.

we show how to instantiate the abstract implementation with a concrete protocol in the Random Oracle Model.

2 Preliminaries

In this section, we introduce the notations we use and recall some standard cryptographic notions and primitives.

2.1 Notations

Let N denote the set of positive integers, and $[n]$ denote the set $\{1, 2, \ldots, n\}$. By a probabilistic algorithm we mean a Turing machine that receives an auxiliary random tape as input. PPT stands for probabilistic polynomial-time algorithms, and nuPPT denotes non-uniform probabilistic polynomial-time algorithms. If M is a probabilistic algorithm, then for any input x, the notation "$M_r(x)$" or "$M(x;r)$" denotes the output of the M on input x when M's random tape is fixed to r, while $M(x)$ represents the distribution of outputs of $M_r(x)$ when r is chosen uniformly. An oracle algorithm is a machine that gets oracle access to another machine. Given a probabilistic oracle algorithm M and a probabilistic algorithm A, we let $M^A(x)$ denote the probability distribution over the outputs of the oracle algorithm M on input x, when given oracle access to A.

By $x \leftarrow \mathcal{S}$, we denote an element x is sampled from a distribution $\mathcal{S}$. If F is a finite set, then $x \leftarrow F$ means x is sampled uniformly from the set F . To denote the ordered sequence in which the experiments happen we use semicolon, e.g. $(x \leftarrow \mathcal{S}; (y,z) \leftarrow A(x))$. Using this notation we can describe probability of events. For example, if $p(\cdot,\cdot)$ denotes a predicate, then $Pr[x \leftarrow \mathcal{S}; (y,z) \leftarrow A(x) : p(y,z)]$ is the probability that the predicate $p(y,z)$ is true in the ordered sequence of experiments $(x \leftarrow \mathcal{S}; (y,z) \leftarrow A(x))$. The notation $\{(x \leftarrow \mathcal{S}; (y,z) \leftarrow A(x) : (y,z))\}$ denotes the resulting probability distribution $\{(y,z)\}$ generated by the ordered sequence of experiments $(x \leftarrow \mathcal{S}; (y,z) \leftarrow A(x))$.

If A and B are probabilistic *interactive* algorithms, we denote by "$A \leftrightarrow B$" the interaction between A and B. This interaction is formally a random variable describing the joint view of both parties (which includes both parties' inputs, random tapes, and messages received). Given an outcome (i.e., joint view) o of an interaction, we let

- $\mathsf{trans}(o)$ denote the transcript of the interaction in o (i.e., the messages sent by the players);
- $\mathsf{view}_b(o)$ denote the view of the b'th player in o;
- $\mathsf{out}_b(o)$ denote the output of the b'th player in o;
- $\mathsf{out}(o)$ denote the output of both players in o.

2.2 Computational Indistinguishability

We recall the definition of computational indistinguishability [23].

Definition 1 (Computational Indistinguishability) *Let X and Y be countable sets. Two ensembles $\{A_{x,y}\}_{x\in X,y\in Y}$ and $\{B_{x,y}\}_{x\in X,y\in Y}$ are said to be computationall indistinguishable if for every probabilistic distinguishing algorithm D with polynomial running-time in its first input, there exists a negligible function $\mu(\cdot)$ such that for every $x \in X, y \in Y$ it holds that:*

$$\left|\Pr\left[a \leftarrow A_{x,y} : D(x,y,a) = 1\right] - \Pr\left[b \leftarrow B_{x,y} : D(x,y,b) = 1\right]\right| < \mu(|x|)$$

2.3 Signatures Schemes

We recall the definition of signature schemes that are secure against adaptive chosen message attacks [24].

Definition 2 (Signature Schemes) *We say that a tuple of probabilistic polynomial-time algorithms $\pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ is a* signature scheme *if the following conditions hold:*

- **Validity:** *For every $n \in \mathbb{N}$, every $(pk, sk) \in \mathsf{Gen}(1^n)$, every $m, s \in \{0,1\}^n$, every $\sigma \in \mathsf{Sign}(\mathsf{sk}, m)$, $\mathsf{Ver}(\mathsf{vk}, m, \sigma) = 1$.*
- **Unforgeability:** *For every nuPPT A, there exists a negligible function μ such that for every $n \in \mathbb{N}$, the probability that A wins in the following experiment is bounded by $\mu(n)$: Sample $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$ and let $A(1^n, \mathsf{vk})$ get oracle access to $\mathsf{Sign}(\mathsf{sk}, \cdot)$; A is said to win if it manages to output a message-signature pair (m, σ) such that $\mathsf{Ver}(\mathsf{vk}, m, \sigma) = 1$ while never having queried its oracle on m.*

Signature schemes can be constructed based on any one-way functions [25, 26].

2.4 Pseudorandom Functions

We recall the definition of a pseudorandom function (PRF).

Definition 3 (PRF) *We say that a PPT computable function f is a* pseudorandom function *over $\{R_n\}_{n\in\mathbb{N}}$ if for every $n \in \mathbb{N}, s \in \{0,1\}^n$, $f(s,\cdot)$ is a function from $\{0,1\}^n$ to R_n, and for every nuPPT oracle machine D there exists a negligible function $\mu(\cdot)$ such that*

$$\left|\Pr\left[\rho \leftarrow \{\{0,1\}^n \to R_n\} : D^{\rho}(1^n) = 1\right] - \Pr\left[s \leftarrow \{0,1\}^n : D^{f(s,\cdot)}(1^n) = 1\right]\right| < \mu(n)$$

PRFs over $\{\{0,1\}^n\}_{n\in\mathbb{N}}$ can be constructed based on any one-way functions [27, 28].

2.5 Commitment Schemes

Commitment protocols allow a *sender* to commit itself to a value while keeping it secret from the *receiver*; this property is called hiding. At a later time, the commitment can only be opened to a single value as determined during the commitment protocol; this property is called binding. For our purposes, we restrict our attention to *collections of non-interactive commitments*.

Definition 4 (Collection of Non-interactive Commitments) *We say that a tuple of PPT algorithms* $(\mathsf{Gen}, \mathsf{Com})$ *is a* collection of non-interactive commitments *if the following conditions hold:*

- **Computational Binding:** *For every nuPPT A there exists a negligible function μ such that for every $n \in \mathbb{N}$,*

$$\Pr\left[\begin{array}{c} i \leftarrow \mathsf{Gen}(1^n) : (m_0, r_0), (m_1, r_1) \leftarrow A(1^n, i) \ : \\ m_0 \neq m_1, |m_0| = |m_1| = n, \mathsf{Com}_i(m_0; r_0) = \mathsf{Com}_i(m_1; r_1) \end{array}\right] \leq \mu(n)$$

- **Computational Hiding:** *The following ensembles are computationally indistinguishable:*
  - $\{Com_i(m_0)\}_{n\in\mathbb{N}, i\in\{0,1\}^*, m_0, m_1\in\{0,1\}^n, , z\in\{0,1\}^*}$
  - $\{Com_i(m_1)\}_{n\in\mathbb{N}, i\in\{0,1\}^*, m_0, m_1\in\{0,1\}^n, z\in\{0,1\}^*}$

Collections of non-interactive commitments can be constructed based on any one-way function [29, 28].[7]

3 Ad-hoc Surveys

An Ad-hoc Survey Scheme is a protocol involving three types of players:

— A *single* Registration Authority (RA).

— One or multiple Survey Authorities (SA).

— Users; each user is associate with a *public* user identity *id* (e.g., its email address).

We assume that the RA has the ability to set up a secure session (private and authenticated) with the user associated with a particular user identity. Each user additionally has the ability to setup an anonymous connection to the SA when returning their survey.

An *ad-hoc survey scheme* is a tuple of algorithms $(\mathsf{GenRA}, \mathsf{RegUser}^{\mathsf{RA}}, \mathsf{RegUser}^{U}, \mathsf{GenSurvey}, \mathsf{Authorized}, \mathsf{SubmitSurvey}, \mathsf{Check})$ which we formalize shortly. To gain some intuition, let us first explain how these algorithms are intended to be used in a system and informally explain what types of security requirements are needed from them.

[7] In fact, those commitments satisfy an even stronger "statistical" binding property which quantifies over all (potentially unbounded) attackers A.

System Set-up

— The RA generates a public key-pair $\mathsf{vk}_{\mathsf{RA}}, \mathsf{sk}_{\mathsf{RA}} \leftarrow \mathsf{GenRA}(1^n)$; $\mathsf{vk}_{\mathsf{RA}}$ is made public and $\mathsf{sk}_{\mathsf{RA}}$ is secretly stored by the RA.

— For systems that require the use of a Common Reference String (CRS), a CRS is generated and made publicly available. For simplicity of notation, we omit the CRS in all the procedures below and simply assume that all these procedures get the CRS as an input. Likewise, for systems in the Random Oracle model, we assume the procedures below have access to the Random Oracle.

User Registration To use the system, users need to register with the RA; at this point the user and the RA execute the protocol $(\mathsf{RegUser}^{\mathsf{RA}}, \mathsf{RegUser}^{U})$ which allows the user to check out an *unlinkable "master credential"*. A user with identity *id* proceeds as follows:

1. The user sets up a secure session with the RA.
2. The RA checks that user identity *id* previously has not been registered. If it has, the RA closes the session. Otherwise, the RA and the user invoke the interactive protocol $(\mathsf{RegUser}^{\mathsf{RA}}, \mathsf{RegUser}^{U})$ on the common input $1^n, id$.
3. If the protocol ends successfully, the RA stores that user identity *id* has been registered, and the user secretly stores the output as cred_{id}.

Survey Registration Whenever an SA wants to set-up a survey with identifier vid, it generates a *survey public-key* based on the identities of the participants. More precisely, the SA on input a survey identifier vid and a list L of user identities (they may be previously registered or not) computes and makes public

$$\mathsf{vk}_{\mathsf{vid}} \leftarrow \mathsf{GenSurvey}(1^n, \mathsf{vid}, L)$$

Completing a Survey Given a registered survey with identifier vid and its associated public-key $\mathsf{vk}_{\mathsf{vid}}$, each authorized user *id* can combine its master credential cred_{id} with the survey identifier vid and public-key $\mathsf{vk}_{\mathsf{vid}}$ to generate an unlikable *one-time token* that it can then use to make a submission in the survey. Roughly speaking, the one-time token satisfies two properties: 1) it carries no link to the students identity *id* (thus we have anonymity), and 2) for a given survey key, the student can obtain at most one such token (and thus can only submit one response).

More precisely, user *id* with master credential cred_{id} submits the message m as the completed survey by privately executing the algorithm

$$\mathsf{Sub} = (\mathsf{tok}, m, \mathsf{tokauth}) \leftarrow \mathsf{SubmitSurvey}(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, m, \mathsf{cred}_{id})$$

and then submitting Sub to the SA through an anonymous channel; tok is the one-time token, and tokauth is an authenticator required to bind the message m to the one-time token, and to ensure uniqueness of the one-time token. SA checks whether the submission is correctly computed by executing

$$\mathsf{Check}(\mathsf{vk}_{\mathsf{RA}}, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, \mathsf{Sub})$$

If it outputs accept it stores the submission. If a submission with the same token tok has been previously stored (i.e., if a Sub of the form $(\mathsf{tok}, m', \mathsf{tokauth}')$ has already been stored, the old record is removed. (Or alternatively, the new Sub is not stored.))

Announcing the results Once all the submissions have been collected, the SA may (depending on external privacy requirements) publish a list of all stored submissions $\mathsf{Sub} = (\mathsf{tok}, m, \mathsf{tokauth})$.

Audit Procedures The system also includes audit procedures. First, users can check that their submission was recorded by simply inspecting that their submission is output. Second, a user may use $\mathsf{Check}(\mathsf{vk}_{\mathsf{RA}}, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, \mathsf{Sub})$ to check whether Sub is a valid submission (i.e., user can check that there is no "ballot/survey-stuffing"). Finally, to ensure that a survey is not targeted to a particular user (for de-anonymization purposes), the user may use function $\mathsf{Authorized}(\mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, id')$ to check whether user id' is also authorized for survey vid with public key $\mathsf{vk}_{\mathsf{vid}}$.

Key features and Security Properties A crucial aspect of an ad-hoc survey is the *privacy* property: even if the RA and SA are arbitrarily corrupted (and in collusion) they cannot learn anything about how particular users answered submissions. The key *security* property of our ad-hoc survey is that only authorized users can complete a survey, and furthermore they can complete it at most *once*.

3.1 Definition of an Ad-hoc Survey

We proceed to provide a formal definition of Ad-hoc Survey schemes and their privacy and security properties.

Definition 5 *An* ad-hoc survey scheme Γ *is a tuple of PPT algorithms and interactive PPTs* $(\mathsf{GenRA}, \mathsf{RegUser}^{\mathsf{RA}}, \mathsf{RegUser}^{U}, \mathsf{GenSurvey}, \mathsf{Authorized}, \mathsf{SubmitSurvey}, \mathsf{Check})$ *where*

- $\mathsf{GenRA}(1^n)$ *outputs a key-pair* $\mathsf{vk}_{\mathsf{RA}}, \mathsf{sk}_{\mathsf{RA}}$.
- $\mathsf{RegUser}^{\mathsf{RA}}(1^n, \mathsf{sk}_{\mathsf{RA}}, \mathsf{vk}_{\mathsf{RA}}, id)$ *is an interactive PPT that outputs either* accept *or* fail.
- $\mathsf{RegUser}^{U}(1^n, \mathsf{vk}_{\mathsf{RA}}, id)$ *is an interactive PPT that outputs a bitstring* cred_{id} *or* fail.

- $\mathsf{GenSurvey}(1^n, \mathsf{vid}, L)$ *outputs a bitstring* $\mathsf{vk}_{\mathsf{vid}}$. *Here* vid *is a unique arbitrary identifier and* L *is a description of the set of users eligible to participate in the survey.*
- $\mathsf{Authorized}(\mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, id)$ *outputs either* accept *or* fail.
- $\mathsf{SubmitSurvey}(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, m, \mathsf{cred}_{id})$ *outputs* $\mathsf{Sub} = (\mathsf{tok}, m, \mathsf{tokauth})$.
- $\mathsf{Check}(\mathsf{vk}_{\mathsf{RA}}, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, \mathsf{Sub})$ *outputs either* accept *or* fail.

A remark on the Authorized procedure We are interested in schemes where the description of the authorized procedure makes it possible to naturally interpret the *set* of users that are allowed to complete a survey (and indeed, our constructions fall into this category). For instance, the description of the Authorized procedure specifies a list of user identities, or specifies a list of user identities with wildcard (e.g., $*@*.cornell.edu$). In our specific implementation, the public key for the survey $\mathsf{vk}_{\mathsf{vid}}$ consists of a list of authorized users.

3.2 Correctness

We proceed to define what it means for an ad-hoc survey scheme to be *correct*. The following definition requires that for every set of users L, if an SA sets up a survey for L, and if a user in L correctly registers with the RA, then this user will be authorized to complete the survey, and will be able to submit a response that passes the check. Additionally, we require all (authorized) users' submissions have different token numbers (or else some of the submission would be thrown out).

Definition 6 *An ad-hoc survey scheme* Γ *is* correct *if there exists a negligible function* $\mu(\cdot)$*, such that the following experiment outputs* fail *with probability at most* $\mu(n)$ *for every* $n \in \mathbb{N}$*,* $\mathsf{vid}, m \in \{0,1\}^n$*, set* L *of* n*-bit strings, every mapping* $m : L \to \{0,1\}^n$*:*

- $(\mathsf{vk}_{\mathsf{RA}}, \mathsf{sk}_{\mathsf{RA}}) \leftarrow \mathsf{GenRA}(1^n)$
- *For every* $id \in L$*:*
  - $(out_{\mathsf{RA}}, out_{\mathsf{U}}) \leftarrow \mathsf{out}[\mathsf{RegUser}^{\mathsf{RA}}(1^n, \mathsf{sk}_{\mathsf{RA}}, \mathsf{vk}_{\mathsf{RA}}, id) \leftrightarrow \mathsf{RegUser}^{U}(1^n, \mathsf{vk}_{\mathsf{RA}}, id)]$
  - *Output* fail *if either* out_{RA} *or* out_{U} *is* fail*; otherwise let* $\mathsf{cred}_{id} = out_{\mathsf{U}}$.
- $\mathsf{vk}_{\mathsf{vid}} \leftarrow \mathsf{GenSurvey}(1^n, \mathsf{vid}, L)$
- *Output* fail *if there exists some* $id \in L$ *such that* $\mathsf{Authorized}(\mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, id) = \mathsf{fail}$
- *For every* $id \in L$*:*
  - $\mathsf{Sub}_{id} = (\mathsf{tok}_{id}, m_{id}, \mathsf{tokauth}_{id}) \leftarrow \mathsf{SubmitSurvey}(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, m[id], \mathsf{cred}_{id})$.

– *Output* fail *if* $\mathsf{Check}(\mathsf{vk}_{\mathsf{RA}}, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, \mathsf{Sub}_{id}) = \mathsf{fail}$, *or if* $m_{id} \neq m[id]$.

– *Output* fail *if there exists* $id, id' \in L$ *such that* Sub_{id} *and* $\mathsf{Sub}_{id'}$ *have the same first component (i.e., the same token number).*

3.3 Privacy and Security

We turn to defining the privacy and security requirements of an ad-hoc survey scheme.

- *Anonymity (Unlinkability).* Roughly speaking, the privacy property stipulates that nobody, even the SAs and RA *in collusion*, can link a specific submission to a specific user. This property holds, even if the link between users and submissions in other surveys are revealed to the attacker.

- *Authenticity (Security).* This propety ensures a (potentially) malicious user can only complete surveys they are authorized for, and additionally can only complete such surveys once (or more precisely, if they successfully submit multiple times, their submissions all use the same token-number and can be easily identified and joined, or discarded depending on the survey policy).

3.3.1 Anonymity (Unlikability)

The following definition stipulates that the SA(s) and RA and malicious users, even if arbitrarily corrupted (and in collusion), cannot distinguish the submissions of two authorized honest users, even for an adaptively chosen participant list, user identities and submission messages, and even if they may see the submission messages of the two participants for any messages of their choosing, in any *other* survey of its choice. Thus, even if an attacker knows what submissions correspond to which users in any surveys of its choice (before and after the survey of interest), it still cannot identify what these users submitted for a survey of interest. The definition mirrors the definition of CCA-secure encryption: we give the attacker the opportunity to generate an arbitrary public-key for the RA, pick two user identities id_0, id_1, ask the users to register with him, and then make oracle queries to the users' SubmitSurvey procedure. Finally, the attacker selects a survey consisting of a vid and a public key for the survey $\mathsf{vk}_{\mathsf{vid}}$ such that id_0, id_1 are both authorized (for which it has not yet queried the SubmitSurvey oracle on vid), a pair of messages m_0, m_1, and then sees two submissions. The attacker must guess whether the two submissions correspond to ones from id_0 and id_1 or from id_1 and id_0 respectively; the attacker continues to have oracle access to the users' SubmitSurvey procedure during this decision-making phase but cannot make queries on the vid.

Definition 7 *An ad-hoc survey scheme* Γ *is* unlinkable *if for every non-uniform PPT* A *the ensembles* $\{\mathsf{EXEC}^0(1^n, A)\}_{n\in\mathbb{N}}$ *and* $\{\mathsf{EXEC}^1(1^n, A)\}_{n\in\mathbb{N}}$ *are computationally indistinguishable where* $\mathsf{EXEC}^b(1^n, A)$ *is defined as follows:*

– $(\mathsf{vk}_{\mathsf{RA}}, z) \leftarrow A(1^n)$

– Let adversary $A(1^n)$ concurrently interact with $\mathsf{RegUser}^U(1^n, \mathsf{vk}_{\mathsf{RA}}, \cdot)$ with adaptively chosen, but unique, user identities $id \in \{0,1\}^n$ of its choice. (That is, A can only use each user identity id in a single interaction with $\mathsf{RegUser}^U$.) Whenever an interaction with some $\mathsf{RegUser}^U(1^n, \mathsf{vk}_{\mathsf{RA}}, id)$ successfully completes (i.e., the output of the user is not fail), let cred_{id} be the output of $\mathsf{RegUser}^U$, and for the remainder of the experiment, A gets oracle access to $\mathsf{SubmitSurvey}(1^n, \cdot, \cdot, \cdot, \mathsf{cred}_{id})$.

– $(\mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, id_0, id_1, m_0, m_1, z') \leftarrow A(1^n, z)$

– Output fail if $\mathsf{Authorized}(\mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, id_\beta) = \mathsf{fail}$ for any $\beta \in \{0,1\}$.

– Let $\mathsf{Sub}_\beta = \mathsf{SubmitSurvey}(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, m_\beta, \mathsf{cred}_{id_{\beta \oplus b}})$ for $\beta \in \{0,1\}$.

– $out \leftarrow A(1^n, (\mathsf{Sub}_0, \mathsf{Sub}_1), z')$

– Output fail if there exists some $\beta \in \{0,1\}$ such that A queried its $\mathsf{SubmitSurvey}(1^n, \cdot, \cdot, \cdot, \mathsf{cred}_{id_\beta})$ oracle with vid as its second input; otherwise, output out.

3.3.2 Authenticity (Security)

Let us now turn to defining security. The following definition stipulates that only authorized users may complete surveys, and only one of their submissions is counted. We require that this holds even if the attacker may register multiple identities, and see submissions of the attacker's choice for any other user of its choice and in any survey (this one, or any other survey).

To begin, we formalize what it means to give the attacker access to submission oracles for users of its choice by defining the stateful oracle

$$\mathsf{SubmitSurvey}'(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, m, id, \mathsf{vk}_{\mathsf{RA}}, \mathsf{sk}_{\mathsf{RA}})$$

that operates as follows: if the oracle has not previously been queried on the identity id (as its 5th input), let $(out_{\mathsf{RA}}, \mathsf{cred}_{id})$ be the output of an execution of

$$\mathsf{RegUser}^{\mathsf{RA}}(\mathsf{sk}_{\mathsf{RA}}, 1^n, \mathsf{vk}_{\mathsf{RA}}, id) \leftrightarrow \mathsf{RegUser}^U(1^n, \mathsf{vk}_{\mathsf{RA}}, id)$$

Next output $\mathsf{SubmitSurvey}(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, m, \mathsf{cred}_{id})$. If the oracle has previously been queried on the identity id, recover the previously computed credential cred_{id}, and directly output $\mathsf{SubmitSurvey}(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, m, \mathsf{cred}_{id})$.

Definition 8 *An ad-hoc survey scheme Γ is* secure against malicious users *if for every non-uniform PPT A, every polynomial $p(\cdot)$, there exists a negligible function $\mu(\cdot)$, such that the following experiment outputs* accept *with probability at most $\mu(n)$ for every $n \in \mathbb{N}$,*

– $(\mathsf{vk}_{\mathsf{RA}}, \mathsf{sk}_{\mathsf{RA}}) \leftarrow \mathsf{GenRA}(1^n)$

– **Honest User Registration and Survey Submission:** Throughout the experiment, $A(1^n)$ has oracle access to $\mathsf{SubmitSurvey}'(1^n, \cdot, \cdot, \cdot, \mathsf{vk}_{\mathsf{RA}}, \mathsf{sk}_{\mathsf{RA}})$. *Let L_{honest} be the set of identities on which A queries the oracle (as its 5th input).*

– **Corrupt User Registration:** Throughout the experiment, A can concurrently interact with $\mathsf{RegUser}^{\mathsf{RA}}(1^n, \mathsf{sk}_{\mathsf{RA}})$ with adaptively chosen, but unique, user identities $id \in \{0,1\}^n$ of its choice. (That is, A can only use each user identity id in a single interaction with $\mathsf{RegUser}^{\mathsf{RA}}$.) *Let $L_{corrupt}$ be the set of identities on which* $\mathsf{RegUser}^{\mathsf{RA}}$ *outputs* accept.

– $z, L, \mathsf{vid} \leftarrow A(1^n)$

– $\mathsf{vk}_{\mathsf{vid}} \leftarrow \mathsf{GenSurvey}(1^n, \mathsf{vid}, L)$

– $S \leftarrow A(1^n, z, \mathsf{vk}_{\mathsf{vid}})$.

– *Output* accept *if*

1. $L_{honest} \cap L_{corrupt} = \emptyset$;
2. $|S| > |L \cap L_{corrupt}|$;
3. $\mathsf{Check}(pk_{\mathsf{RA}}, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, \mathsf{Sub})$ *accepts for all* $\mathsf{Sub} \in S$;
4. *for any two* $(\mathsf{tok}, m, \mathsf{tokauth})$, $(\mathsf{tok}', m', \mathsf{tokauth}') \in S$, $\mathsf{tok} \neq \mathsf{tok}'$.
5. *For all* $(\mathsf{tok}, m, \mathsf{tokauth}) \in S$, $(\mathsf{tok}, m, \mathsf{tokauth}')$ *was never received as an output from A's* $\mathsf{SubmitSurvey}'(1^n, \mathsf{vid}, \cdot, m)$ *oracle.*

The above four conditions correspond to the determining whether 1) A produced more submissions than it is authorized to do, 2) all submissions are valid (or else they would be rejected), 3) all submissions have different token-numbers (or else only one of them is counted), and 4) all token-numbers are "new" (i.e., it didn't "take" them from some honest user for which it has seen a response).

To get some intuition for what these conditions imply, note that a minimal property guaranteed by these conditions (and the correctness property of ad-hoc surverys) is that if a user submits multiple responses, they will all receive the same token number tok (and thus only one of them will be counted). Additionally, these conditions guarantee that a malicious user, even if it can see responses of its choice of other users (which thus have different token numbers), and even if they by mistake submit many responses, it still cannot come up with a valid submission with a *new* token-number. Finally, the fourth condition is worth elaborating on: Not only it guarantees that an attacker cannot come up with a new token number tok, but also that it cannot "change" the response of a user, even if this user has filled out the same survey multiple times: we allow the

attacker to request many submissions by some honest user, which thus all will have the same token-number tok, on (perhaps different) messages of its choice, yet the attacker should still not be able to come up with a valid submission $(\mathsf{tok}, m, \mathsf{tokauth})$ for some *new* message m.

Remark 1 *In the above definition, we restrict the attacker to only output a* single *"challenge". survey. However, this is without loss of generality: any A that breaks security in one out of $p(n)$ surveys, can be used to construct an attacker A' that picks a random of the surveys as its "challenge" survey and internally emulates all the others. This only decreases A success probability by at most a factor $1/p(n)$.*

Remark 2 *An even stronger definition of security would require that whenever the attacker makes a submission, it must "know" which of its registered and authorized users it is actually submitting for. More precisely, this requires the existence of an "online" extractor that extracts out the identity for each valid submission. We mention that our scheme also satisfies this "explicit knowledge" notion.*

4 An Ad-hoc Survey Scheme Based on General Assumptions

In this section we provide a general construction of Ad-hoc Survey systems. As described in the introduction, this construction will be based on general assumptions and primitives that later can be instantiated with concrete protocols. In particular, as shown in Section 5, the general protocol can be instantiated under concrete assumption in the Random Oracle Model. (Alternatively, these primitives can be instantiated under general assumptions in the CRS model, but the resulting solution would no longer be practical.)

Towards providing our protocol, we start by introducing and studying two new primitives: *online simulation-extractable NIZK* and *partially blind signatures*.

4.1 Online Simulation-Extractable NIZK

We consider a notion of *"online" concurrent simulation-extractable (oSE) NIZK*, a non-interactive version of tag-based concurrent simulation-extractable zero-knowledge from [12, 13], but with an *online* simulation-extraction property à la definition of *universally composable UC NIZK* of [30]. (This notion is closely related—but stronger than—the notions of *non-malleability in the explicit witnesses sense* of [31] (which in turn relies on the notion of *simulation-soundness* of [11]).) We note that our definition is essentially equivalent to the notion of UC NIZK, but to simplify exposition (and for self-containment), we prefer to provide an explicit definition without needing to import the UC framework.

Since our instantiated protocol will be in either the Random Oracle Model (ROM) or in the Common Reference String (CRS) model, we provide a definition of *oracle NIZK*, where all parties have access to some auxiliary oracle $\mathcal{O} = \{\mathcal{O}_n\}_{n\in\mathbb{N}}$ where $\mathcal{O}_n$ is a

distribution over functions $\{0,1\}^{p(n)} \to \{0,1\}^{p'(n)}$ for polynomials p, p'. In the Random Oracle model, we consider a setting where all parties have access to $\mathbb{RO}_{p,p'} = \{\mathbb{O}_n\}_{n\in\mathbb{N}}$ where $\mathbb{O}_n$ is simply uniform distribution over functions $\{0,1\}^{p(n)} \to \{0,1\}^n$. In the CRS model, on the other hand, we consider a degenerate oracle that simply outputs a single string (namely the CRS). For our purposes, we restrict to oracles $\mathbb{O}_n$ that can be efficiently implemented (e.g., it is well known that a Random oracle can be implemented in PPT using lazy evaluation).

For our purposes (and following [12, 13]) we require the use of a "tag-based" NIZK in which both the prover and the verifier get an additional string tag as input.[8]

Definition 9 (Tag-Based Non-Interactive Proofs) *A tuple $(P, V, \mathbb{O})$, is called a tag-based non-interactive proof system for a language L with witness-relation $R_L(\cdot)$ if the algorithm V is a deterministic polynomial-time, and P is probabilistic polynomial-time, and $\mathbb{O} = \{\mathbb{O}_n\}_{n\in\mathbb{N}}$ is a sequence of distributions $\mathbb{O}_n$ over oracles $\{0,1\}^{p(n)} \to \{0,1\}^{p'(n)}$ (where p, p' are polynomials), such that the following two conditions hold:*

- **Completeness:** *There exists a negligible function μ such that for every $n \in \mathbb{N}$, $x \in L$ s.t. $|x| = n$, every $w \in R_L(x)$ and every $\mathsf{tag} \in \{0,1\}^n$,*

$$\Pr[\ \rho \leftarrow \mathbb{O}_n;\ \pi \leftarrow P^\rho(\mathsf{tag}, x, w)\ :\ V^\rho(\mathsf{tag}, x, \pi) = 1\ ] \geq 1 - \mu(n)$$

- **Soundness:** *For every algorithm B, there exists a negligible function μ such for every $n \in \mathbb{N}$, $x \notin L$ s.t. $|x| = n$ and every $\mathsf{tag} \in \{0,1\}^n$,*

$$\Pr[\ \rho \leftarrow O_n;\ \pi' \leftarrow B^\rho(\mathsf{tag}, x)\ :\ V^\rho(\mathsf{tag}, x, \pi') = 1\ ] \leq \mu(n)$$

If the soundness condition only holds for polynomial-time adversaries B, then (P, V) is a non-interactive argument.

Before defining our notion of simulation-extractability, let us first define an intermediary notion which only strengthens the soundness condition of non-interactive arguments to an online extraction condition.

Definition 10 (Tag-Based Online Extractable Non-Interactive Arguments) *A tag-based non-interactive argument $(P, V, \mathbb{O})$ (resp. tag-based non-interactive proof system) for a language L with witness-relation $R_L(\cdot)$ is called* online extractable *if the following condition holds:*

- **Online Extraction:** *There exists a PPT extractor X such that for any nuPPT algorithm B, there exists a negligible function μ such for every $n \in \mathbb{N}$,*

$$\Pr\left[\begin{array}{c}\rho \leftarrow O_n;\ \mathsf{tag}, x, \pi' \leftarrow B^\rho(1^n)\ : \\ V^\rho(\mathsf{tag}, x, \pi') = 1 \wedge \mathsf{tag} \in \{0,1\}^n \wedge X(x, Q) \notin R_L(x)\end{array}\right] \leq \mu(n)$$

where Q is the set of queries, and answers to those queries, made by B to ρ.

[8] The tag is the analog of a "session id" in the UC security setting.

We now turn to defining (concurrent) simulation extractability. Consider a *man-in-the-middle* attacker A that receives proofs from an honest prover "on the left" and outputs its own proofs "on the right". More specifically, in the left interaction, attacker A can request proofs of any true statement x of its choice using any tag tag of its choice. In the right interaction, A outputs a list of tags, statements and proofs $(\vec{\mathsf{tag}}, \vec{x}, \vec{\pi})$. We require that the all the proofs requested by A can be simulated "online" for A (i.e., without knowing the internals of A and without "rewinding" it), and additionally that witnesses for all accepting proofs $(x, \pi) \in (\vec{x}, \vec{\pi})$ on the right that use a "new" tag can be extracted out. To enable this task, the simulator-extractor may simulate all oracle queries made by A (and, in particular, may extract witnesses by observing what queries are made).

We proceed with a formal definition. Let $(P, V, \mathbb{O})$ be an online extractable tag-based NIZK for the language L with witness relation R_L, and with the extractor X. Given any PPT attacker A, "witness determining function" W, $n \in N$ and auxiliary input z to A, let $\mathsf{real}(1^n, A, W, z)$ denote the output of the following experiment: Let $\rho \leftarrow \mathbb{O}_n$; give $A^\rho(1^n, z)$ oracle access to $P'(\cdot, \cdot)$ where $P'(\mathsf{tag}, x)$ runs $P^\rho(\mathsf{tag}, x, W(x, view'))$ where $view'$ is A's view up until this query. Finally, output the view of A (which includes the random coins of A, the proofs received by A, and the answers to all ρ queries made by A); for convenience of notation, we assume (without loss of generality) that A runs V^ρ on any proof that it outputs, *right after outputting it*, and thus the view of A suffices to determine whether a proof it outputs is accepting. We call the pair (A, W) *valid* if A never queries its prover oracle on some statement x such that $W(x, view) \notin R_L(x)$ where $view$ is the view of A at the time of the query.

Given a PPT simulator S, we similarly define $\mathsf{sim}(1^n, A, S, z)$: We let $S(1^n)$ interact with $A(1^n, z)$ in a *straight-line fashion*, without rewinding it: in particular, S gets to answer any ρ queries made by A and any proof request made by A (but can only communicate with A once, and cannot rewind A). Finally, output the view of A in this interaction.

Finally, define the predicate $\mathsf{ExtractFail}(view) = 1$ if and only if A, when given the view $view$, ever outputs a proof π of some statement x with respect to tag tag such that a) none of the proofs received by A in the view $view$ use tag tag, b) V^ρ accepts the proof π (recall that we have assumed that the oracle queries needed to verify π are part of the view of A), and c) $X(x, \pi, Q) \notin R_L(x)$, where Q is the set of oracle queries, and answers to those queries, made by A.

Definition 11 (Online Simulation-Extractability) *Let $(P, V, \mathbb{O})$ be an online extractable tag-based NIZK for the language L with witness relation R_L, and with the extractor X. We say that $(P, V, \mathbb{O})$ is* online simulation extractable (oSE) *if there exists a PPT machine S such that for every PPT A, the following two conditions hold:*

- **Simulatability:** *For all PPT algorithms A, W such that (A, W) is a valid, the following ensembles are computationally indistinguishable:*
  - $\{\mathsf{real}(1^n, A, W, z)\}_{n \in \mathbb{N}, z \in \{0,1\}^*}$

– $\{\mathsf{sim}(1^n, A, S, z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$

- **Extractability:** *There exists a negligible function μ such that for every $n \in \mathbb{N}$, every $\ell \in \mathbb{N}$*

$$\Pr[(view \leftarrow \mathsf{sim}(1^n, A, S, z) : \mathsf{extractfail}(view) = 1] \leq \mu(n)$$

Let us finally remark that any "Universally Composable NIZK" [30, 31, 32] in e.g., the CRS model (which can be based on enhanced trapdoor permutations) directly yield an oSE NIZK in the CRS model:

Theorem 1 *Assume the existence of enhanced trapdoor permutations. Then there exists a oSE NIZK in the CRS model for every language in NP.*

In the next section we turn to providing more efficient implementations in the ROM.

4.1.1 Simulation-extractability from HVZK

In this section we show how to transform any 3-round special-sound special Honest-verifier Zero-knowledge (HVZK) [33] satisfying online simulation-extractability into an online simulation-extractable NIZK in the ROM. Let us first recall such proof systems.

Definition 12 ([33]) *A proof system (P, V) is a 3 round special-sound special Honest-verifier Zero-knowledge (HVZK) proof/argument for language L with witness relation R_L if:*

1. **Three-move form:** *(P, V) is a 3-round public coin interactive proof where the 2-round public coin message is of lenght $t(|x|) > 1$ where x is the common input.*
2. **Special soundness:** *There exists a PPT extractor X such that given any two accepting transcripts (a, c, z) and (a, c', z') for the instance x, $X(x, a, c, c', z, z') \in R_L(x)$.*
3. **Special HVZK:** *There exists a polynomial time simulator S such that the following distributions are computationally indistinguishable*
   - $\{\mathsf{trans}[P(x, w) \leftrightarrow V_c(x)]\}_{x\in L,w\in R_L(x),c\in\{0,1\}^{t(|x|)},z\in\{0,1\}^*}$
   - $\{S(x, c)\}_{x\in L,w\in R_L(x),c\in\{0,1\}^{t(|x|)},z\in\{0,1\}^*}$

Note that in the above definition of special honest-verifier zero-knowledge, the auxiliary input z is not given to any algorithms but is present to ensure that indistinguishability holds also w.r.t. nuPPT distinguishers.

Our transformation proceeds in two steps:

- In the first step, we rely on a result by Pass [34] which transforms any 3-round special-sounds special HVZK protocol into an *online extractable* special HVZK protocol in the Random Oracle Model.

- We next apply the "Fiat Shamir heuristic" [**?**] (but additionally apply the Random Oracle to the tag) to turn any 3-round online extractable special HVZK protocol into a online simulation-extractable NIZK in the Random Oracle model.

Towards this, let us first define oracle-based online-extractable special HVZK protocol.

Definition 13 (Online Extractable Special HVZK Interactive Arguments) *A tuple $(P, V, \mathbb{O})$, is called a online extractable special HVZK interactive argument for a language L with witness-relation $R_L(\cdot)$ if P, V are PPTs that communicate in 3-rounds, V is "public coin" and the length of the second message is $t(n)$ for inputs of lenght n, $\mathbb{O} = \{\mathbb{O}_n\}_{n\in N}$ is a sequence of distributions $\mathbb{O}_n$ over oracles $\{0,1\}^{p(n)} \to \{0,1\}^{p'(n)}$ (where p, p' are polynomials), such that the following two conditions hold:*

- **Completeness:** *There exists a negligible function μ such that for every $n \in \mathbb{N}$, $x \in L$ s.t. $|x| = n$, every $w \in R_L(x)$,*

$$\Pr[\ \rho \leftarrow \mathbb{O}_n;\ \mathsf{out}_2[P^\rho(x,w) \leftrightarrow V^\rho(x)] = 1\ ] \geq 1 - \mu(n)$$

- **Online Extraction:** *There exists a PPT extractor X such that for any nuPPT algorithm B, there exists a negligible function μ such for every $n \in \mathbb{N}$,*

$$\Pr\left[\begin{array}{c} \rho \leftarrow \mathbb{O}_n;\ x, z \leftarrow B^\rho(1^n) : \\ \mathsf{out}_2[B^\rho(x,z) \leftrightarrow V^\rho(x)] = 1 \wedge X(x, Q) \notin R_L(x) \end{array}\right] \leq \mu(n)$$

  where Q denotes the set of queries, and answers to those queries, made by B to ρ.

- **Special HVZK:** *There exists a polynomial time simulator S such that for every nuPPT oracle machine D, there exists a negligible function μ such that for every $x \in L, w \in R_L(x), c \in \{0,1\}^{t(|x|)}$,*

$$|\Pr\left[\rho \leftarrow \mathbb{O}_n : D^\rho(\mathsf{trans}[P^\rho(x,w) \leftrightarrow V_c^\rho(x)]) = 1\right] - \Pr\left[\rho \leftarrow \mathbb{O}_n : D^\rho(S(x,c)) = 1\right]| \leq \mu(|x|)$$

Let us remark that our definition of zero-knowledge in the oracle model is somewhat stronger than what is usually required (and also what we require in our definition of simulation extractability): our definition is "non-programmable" [35, 34] in that we do *not* allow the simulator to program the oracle. This property will be useful to ensure that there will be no "conflict" in the simulation when dealing with multiple proofs (which is needed for acheiving simulation extractability).

Step 1: Achieving online extractable special HVZK The following theorem is proven in [34] (using slightly different language).

Theorem 2 *Let (P, V) be a 3-round special-sound special HVZK protocol for L with witness relation R_L. Then, there exists a polynomial p and PPT algorithms P', V' such that $(P', V', \mathbb{RO}_p)$ is an online extractable special HVZK for L with witness relation R_L.*

Additionally, the modified protocol (P', V') is obtained from (P, V) through an efficient transformation (with $\omega(\log n)$ computational overhead).

Step 2: Acheiving simulation-extractable NIZK Given a 3-round online extractable special HVZK protocol $\Pi = (P, V, \mathbb{O})$ where the length of the first message is $p(\cdot)$, and the length of the second message is $t(\cdot)$, we construct a tag-based NIZK $\tilde{P}i = (P', V', (\mathbb{O}, \mathbb{RO}_{\tilde{p},t}))$ where $\tilde{p}(n) = p(n) + n$ by applying the "Fiat-Shamir heuristic", and viewing the tag *tag* as part of the first message—that is, the second-message "challenge" c is generated by applying the random oracle ρ to the first message a and the tag tag (i.e., $c = \rho(a, \mathsf{tag})$), and letting V' accept iff V_c accepts.

Theorem 3 *Let $(P, V, \mathbb{O})$ be an online extractable special HVZK argument for L with witness relation R_L, where the first message a of Π has $\omega(\log n)$ min-entropy [9] and the second message b is of length $\omega(\log n)$ (where n is the length of the common input x). Then $\tilde{\Pi}$ constructed above is a tag-based online simulation-extractable argument.*

Proof: For simplicity of notation, we assume that the challenge b in Π has length $t(n) = n$ (where n is the statement length); the proof easily extends to the case when the challenge is of super-logarithmic length. Let X denote the online extractor guaranteed to exists for Π. Consider some simulation-extractability attacker A' for $\tilde{\Pi}$. Recall that A' asks to hear proofs "on the left" and provides proofs "on the right". We show how to simulate all "left" proofs (and additionally all oracle queries), while ensuring that X still succeeds in extracting witnesses from all accepting proofs on the right (that use a new tag).

Let ρ denote the $\mathbb{O}_n$ oracle, and RO denote the random oracle. The simulator $S(1^n)$ proceeds as follows:

Simulation of ρ queries: S honestly emulates these queries. (Recall that we restrict to oracles $\mathbb{O}_n$ that can be efficiently implemented, and thus this step can be performed efficiently.)

Simulation of $P'(tag, x)$ queries: S picks a random challenge $c \in \{0,1\}^n$, and runs the HVZK simulator for Π^ρ on (x, c) to obtain a transcript (a, c, z)—we here rely on the fact that the simulator does not program ρ, and thus the simulator needs to make ρ queries which (as per above) are honestly emulated. We finally "program" $\mathbb{RO}$ to return c on the input (a, tag) and store this query-answer pair, and return (a, c, z).

[9] Every special-sound special HVZK argument for a hard-on-the-average language must have this property: if not, with polynomial probability two honest executions would have the same first message, but different second messages and thus a witness can be extracted out.

In the unlikely case that S attempts to program the oracle on an input that was already stored, S halts and outputs fail.

Simulation of RO queries: For any RO query that that has not previously been stored, simply pick a random n-bitstring, return it, and store this query-answer pair; if a query-answer pair has been previously stored, simply output the answer.

Correctness of the Simulation Let us first argue that, assuming that S never outputs fail, the simulation is correctly distributed. As usual, we define a sequence of hybrids $H_0, \ldots, H_{m(n)}$ (where $m(n)$ is the running-time of A), where H_i denotes the view of A in a hybrid experiment where the first i queries to P''s responses are simulated and the remaining ones are honestly emulated (but all $\rho, \mathbb{RO}$ queries are still simulated by S). Note that $H_{m(n)}$ is the simulated experiment, and H_0 is identically distributed to the real experiment (this follows from the fact that S honestly emulates ρ and $\mathbb{RO}$ queries if no P' queries are simulated). By a hybrid argument, we need to show that any two consequetive hybrids H_i, H_{i+1} are indistinguishable. Note that the only difference between these two experiments is that the $(i+1)$'th "left" proof is honestly generated simulated in H_i, whereas it is simulated in H_{i+1} (and thus $\mathbb{RO}$ is programmed at one more point in H_{i+1}). Assume for contradiction that there exists a distinguisher D and a polynomial p, such that for infinitely many n, there exists some i such that D distinguishes H_i and H_{i+1} with probability $1/p(n)$, for some polynomial p. Towards reaching a contradiction, consider yet another hybird $\tilde{H}_i$ which is defined identically to H_{i+1}, except that when generating the ith "left" proof, instead of running the HVZK simulator on input of a uniformly generated challenge c to generate the transcript (a, c, z), generate it by letting the honest prover communicate with V_c^ρ. Note that since the challenge c is uniformly chosen, it follows that $\tilde{H}_i$ is identically distributed to H_i. Thus, D must distinguish between $\tilde{H}_i$ and H_{i+1} with probability $1/p(n)$ for infinitely many n. For each such n, there thus exists some prefix τ of the view of A, leading up to the point when A asks to here the ith proof (of some statement x using tag tag), and including the choice of the challenge c, such that D distinguishes $\tilde{H}_i$ and H_{i+1} with probability $1/p(n)$ also conditioned on τ (this follows since $\tilde{H}_i$ and H_{i+1} proceed identically the same up until this point). Additionally, the continuation of the experiments can be efficiently generated given τ; this directly contradicts the special HVZK property of Π.

Next, note that S only outputs fail with negligible probability; this directly follows from the fact that the first message in Π has $\omega(\log n)$ min-entropy so the probability that it collides with any previously programmed $\mathbb{RO}$ queries is negligible; as a result, it follows by a union bound that the probability of ever getting a collision is also negligible. We conclude that the simulated experiment is indistinguishable from the real one.

Correctness of the Extraction Assume for contradiction that the extraction fails. That is, there exists some polynomial $p(\cdot)$ such that for infinitely many n, A succeeds in

outputting a proof for statement x w.r.t. some tag tag for which it does not receive any proofs, yet X fails to extract a witness for x. Assume without loss of generality that A never makes the same ρ or RO queries twice. We construct an attacker B for Π such that X fails to extract a witness from B. B internally emulates the whole simulated experiment for A, but picks a random $\mathbb{RO}$ query of the form tag, a and externally forwards a; upon receiving back the challenge c, it sets the answer to the RO query to be c. If eventually A outputs an accepting proof (a, c, z) w.r.t. tag, externally forward z.

Note that unless the query tag, a had already been "programmed" by the simulator, the view of A in the emulation by B is identical to its view in the simulated experiment (and if it had to be programmed, the returned answer will with high probability be inconsistent with the previously set value). Furthermore, recall that we only program $\mathbb{RO}$ on tag, a if A request a proof w.r.t. tag. Thus conditioned on us picking the "right" query (on which extraction fails, yet A uses a "new" tag), B is perfectly simulating the execution for A, and thus conditioned on us picking the "right" query, extraction using X for B fails. Since the number of queries is polynomially bounded, we conclude that B performs an online extraction for π with inverse polynomial probability. □

4.2 Partially Blind Signatures

We consider a variant of *blind signatures* [**?**] where a user (receiver) R can ask a signer S to sign a tuple (m, s) of messages without revealing s. Additionally, this "partially blind" signature is required to be unforgeable under a (standard) adaptive chosen message attack. We formalize this notion using a weak notion of blinding which requires that no efficient signer S^* can tell what s is, even if it observes whether the receiver got a valid signature or not.[10]

Definition 14 *[Partially Blind Signature] We say that a tuple of probabilistic polynomial-time algorithms* $\pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Blind}, \mathsf{Unblind})$ *is a* partially blind signature *if the following conditions hold:*

- **Validity:** *For every* $n \in \mathbb{N}$, *every* $(\mathsf{vk}, \mathsf{sk}) \in \mathsf{Gen}(1^n)$, *every* $m, s \in \{0,1\}^n$, *every random tape* $r \in \{0,1\}^\infty$, *every* $\sigma \in \mathsf{Unblind}_r(\mathsf{vk}, \mathsf{Sign}(\mathsf{sk}, (m, \mathsf{Blind}_r(\mathsf{vk}, s))))$, $\mathsf{Ver}(\mathsf{vk}, (m, s), \sigma) = 1$.

- **Weak Blinding:** *For every polynomial time* S^*, *the following ensembles are computationally indistinguishable*

  - $\{\mathsf{EXEC}^0(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)\}_{n \in \mathbb{N}, z, \mathsf{vk} \in \{0,1\}^*, m, s_0, s_1 \in \{0,1\}^n}$
  - $\{\mathsf{EXEC}^1(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)\}_{n \in \mathbb{N}, z, \mathsf{vk} \in \{0,1\}^*, m, s_0, s_1 \in \{0,1\}^n}$

[10] More traditional definitions of blinding require that a notion of "unlinkability" holds even if the signer gets to see *actual* signatures recovered by the receiver (and not just whether a valid signature was recoverd)..

where $\mathsf{EXEC}^b(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)$ *is defined as follows:*

- $r \leftarrow \{0,1\}^{poly(n)}$ *and let* $y = (m, \mathsf{Blind}_r(\mathsf{vk}, s_b))$*;*
- $x \leftarrow S^*(1^n, z, y)$*;*
- *Let* $o = \mathsf{Ver}(\mathsf{vk}, (m, s), \mathsf{Unblind}_r(\mathsf{vk}, x))$.
- *Output* $S^*(1^n, z, y, o)$.

- **Unforgeability:** *for every nuPPT* A*, there exists a negligible function* μ *such that for every* $n \in N$*, the probability that* A *wins in the following experiment is bounded by* $\mu(n)$*: Sample* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$ *and let* $A(1^n, \mathsf{vk})$ *get oracle access to* $\mathsf{Sign}(\mathsf{sk}, \cdot)$*;* A *is said to win if it manages to output* ℓ *unique message-signature pairs* $((m, s_i), \sigma_i)_{i \in [\ell]}$ *such that* $\mathsf{Ver}(\mathsf{vk}, (m, s_i), \sigma_i) = 1$ *for* $i \in [\ell]$*, while having made less than* ℓ *oracle queries of the form* $(m, \cdot)$.

4.2.1 A construction based on general assumptions

We here provide a construction of partially blind signatures based on one-way functions using standard cryptographic building blocks.

Theorem 4 *Assume the existence of one-way functions. Then there exists a partially blind signature scheme.*

Proof: Let,

- $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme.
- $(\mathsf{Gen}_{com}, \mathsf{Com})$ be a collection of non-interactive commitments over $\{\{0,1\}^n\}_{n \in N}$, where commitments to n-bit strings have length $s(n)$.

Recall that all these primitives can be constructed based on one-way functions (see Section 2 for details). Define:

- $\mathsf{Gen}'(1^n) : \{j \leftarrow \mathsf{Gen}_{com}(1^n), (\mathsf{vk}, \mathsf{sk}) \leftarrow Gen(1^{n+s(n)}) : (j, \mathsf{vk}), \mathsf{sk}\}$.
- $\mathsf{Sign}'_{\mathsf{sk}}(m, c) = \mathsf{Sign}_{\mathsf{sk}}(m, c)$
- $\mathsf{Blind}'_r((j, \mathsf{vk}), s) = \mathsf{Com}_j(s; r)$
- $\mathsf{Unblind}'_r((j, \mathsf{vk}), \sigma) = (r, \sigma)$
- $\mathsf{Ver}'((j, \mathsf{vk}), (m, s), (r, \sigma)) = \mathsf{Ver}(\mathsf{vk}, (m, \mathsf{Com}_j(s; r)), \sigma)$

We now how the following proposition that implies the theorem.

Proposition 5 *If* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *is a secure signature scheme and* $(\mathsf{Gen}_{com}, \mathsf{Com})$ *is a collection of non-interactive commitment schemes, then* $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}', \mathsf{Blind}', \mathsf{Unblind}')$ *is a partially blind signature scheme.*

Proof: We note that the validity property is immediate. We next show unforgeability and then turn to the weak blinding property.

Unforgeability Assume for contradiction that there exists some nuPPT A, and a polynomial p such that for infinitely many $n \in N$, A makes ℓ signature queries of the form $(m, \cdot)$ but manages to output $\ell + 1$ valid message-signature pairs $(m, s_i), (r_i, \sigma_i)$ (for unique messages s_i We have two cases: a) either all $\mathsf{Com}_j(s_i; r_i)$ are unique, or b) there exists i, i' such that $\mathsf{Com}(s_i, r_i) = \mathsf{Com}(s_{i'}; r_{i'})$. One of these cases must happen with probability $p(n)/2$ for infinitely many n. If it is case a), then we violate unforgeability of $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$, and if it is case b), then we violate the binding property of Com_j for an honestly generated j, which is a contradiction.

Weak Blinding Consider some malicious signer S^*, $z, \mathsf{vk}, m, s_0, s_1$ and the execution of $\mathsf{EXEC}^b(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)$. We first show that, without loss of generality, S^* can ignore the "verdict" o (i.e., whether the receiver accepts or rejects the unblinded signature)—it can perfectly "simulate" this verdict on its own. More specifically, recall that in EXEC^b, we first let

$$y = (m, \mathsf{Blind}_r((j, \mathsf{vk}), s_b)) = (m, Com_j(s_b; r))$$

and provide y to S^*; S^* next computes some answer (presumably a blinded signature) x, and finally S^* gets the back the verdict

$$\begin{aligned} \mathsf{Ver}'((j, \mathsf{vk}), (m, s_b), \mathsf{Unblind}_r((j, \mathsf{vk}), x)) = \\ \mathsf{Ver}'((j, \mathsf{vk}), (m, s_b), (r, x)) = \\ \mathsf{Ver}(\mathsf{vk}, (m, \mathsf{Com}_j(s_b; r)), x) = \\ \mathsf{Ver}(\mathsf{vk}, y, x) \end{aligned}$$

Thus instead of returning a (persumed) signature x to the receiver (to find out if it is accepted), S^* could simply emulate the verdict on its own by computing by $\mathsf{Ver}(\mathsf{vk}, y, x)$.

Thus if EXEC^0 and EXEC^1 are distinguishable, then $\mathsf{Blind}((j, \mathsf{vk}), s_0) = \mathsf{Com}_j(s_0)$ and $\mathsf{Blind}((j, \mathsf{vk}), s_1) = \mathsf{Com}_j(s_1)$ are distinguishable, which contradicts the hiding property $(\mathsf{Gen}_{com}, \mathsf{Com})$. □ □

4.3 The Ad-hoc Survey Construction

We now proceed to describe our construction of an ad-hoc survey scheme (based on general assumptions).

4.3.1 Primitives used

We first describe the underlying primitives we rely on. Let,

- $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme.
- $(\mathsf{Gen}', \mathsf{Ver}', \mathsf{Sign}', \mathsf{Blind}', \mathsf{Unblind}')$ be a partially blind signature scheme.
- $\{f_s\}_{s\in\{0,1\}^*}$ be a family of PRFs.
- Let L be the NP language defined as follows: $(\mathsf{tok}, \mathsf{vid}, \mathsf{vk}_{\mathsf{RA}}, \mathsf{vk}_{\mathsf{SA}}) \in L$ iff there exist strings $s, id, \sigma_s, \sigma_{\mathsf{vid}_{id}}$ such that $\mathsf{Ver}'(\mathsf{vk}_{\mathsf{RA}}, (id, s), \sigma_s) = 1$ and $\mathsf{Ver}(\mathsf{vk}_{\mathsf{SA}}, (\mathsf{vid}, id), \sigma_{\mathsf{vid}_{id}}) = 1$ and $\mathsf{tok} = f_s(\mathsf{vid})$
- Let $(P, V, \mathbb{O})$ be an oSE NIZK protocols for L.

4.3.2 The Abstract Ad-hoc Survey Scheme

We are now ready to describe our ad-hoc survey scheme.

- $\mathsf{GenRA}(1^n) = \mathsf{Gen}'(1^n)$;
- $(\mathsf{RegUser}^{\mathsf{RA}}(\mathsf{sk}_{\mathsf{RA}}), \mathsf{RegUser}^{U})(1^n, id_i)$ proceeds as follows:
  - A user with identity id uniformly generates and stores $s \leftarrow \{0,1\}^n, r \leftarrow \{0,1\}^{\mathrm{poly}(n)}$ and computes $y = (id, \mathsf{Blind}_r(\mathsf{vk}_{RA}, s)$ and sends it to the RA.
  - The RA returns $x = \mathsf{Sign}'(\mathsf{sk}_{\mathsf{RA}}, y)$.
  - The user computes $\sigma_s = \mathsf{Unblind}'_r(\mathsf{vk}_{RA}, x)$, checks if $\mathsf{Ver}'(\mathsf{vk}_{RA}, (id, s), \sigma_s) = 1$ and if so outputs $cred = (s, \sigma_s)$ and otherwise $\perp$.
- $\mathsf{GenSurvey}(1^n, \mathsf{vid}, L)$ proceeds as follows.
  - $\mathsf{vk}_{\mathsf{SA}}, \mathsf{sk}_{\mathsf{SA}} \leftarrow \mathsf{Gen}(1^n)$ For each $id \in L$, compute $\sigma^{\mathsf{vid}}_{id} = \mathsf{Sign}(\mathsf{sk}_{SA}, (\mathsf{vid}, id))$ and output $\mathsf{vk}_{\mathsf{SA}}$ and the list of tuples $(id, \sigma^{\mathsf{vid}}_{id})_{id\in L}$.
  - $\mathsf{Authorized}(\mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, id)$ parses $\mathsf{vk}_{\mathsf{vid}}$ as $(\mathsf{vk}_{SA}, \tilde{L})$ and outputs YES iff $\tilde{L}$ contains a record of the form $(id, \sigma^{\mathsf{vid}}_{id})$ such that $\mathsf{Ver}(\mathsf{vk}_{SA}, (\mathsf{vid}, id), \sigma^{\mathsf{vid}}_{id}) = 1$
  - $\mathsf{SubmitSurvey}(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, m, id, \mathsf{cred})$ proceeds as:
    - Parse $\mathsf{cred} = (s, \sigma_s)$.
    - Parses $\mathsf{vk}_{\mathsf{vid}} = (\mathsf{vk}_{SA}, \tilde{L})$
    - Compute token-number $\mathsf{tok} = f_s(\mathsf{vid})$.
    - Recover a tuple of the form $(id, \sigma^{\mathsf{vid}}_{id})$ from $\tilde{L}$. If no such tuple exists, or if $\mathsf{Ver}(\mathsf{vk}_{SA}, (\mathsf{vid}, id), \sigma^{\mathsf{vid}}_{id}) \neq 1$ abort.

* Compute a on SE NIZK π using P that $(\mathsf{tok}, \mathsf{vid}, \mathsf{vk}_{\mathsf{RA}}, \mathsf{vk}_{\mathsf{SA}}) \in L$ with tag $\mathsf{tok}||\mathsf{vid}||m$ using $s, \mathsf{vid}, \sigma_s, \sigma_{id}^{\mathsf{vid}}$ as witness.
* Send the tuple $\mathsf{Sub} = (\mathsf{tok}, m, \pi)$ to the SA.

– $\mathsf{Check}(\mathsf{vk}_{\mathsf{RA}}, \mathsf{vid}, \mathsf{tok}, m, \pi)$ outputs accept if V accepts π as a proof of the statement $(\mathsf{tok}, \mathsf{vid}, \mathsf{vk}_{\mathsf{RA}}) \in L$ with tag $\mathsf{tok}||\mathsf{vid}||m$.

Theorem 6 *If* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *is a secure signature scheme,* $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}', \mathsf{Blind}', \mathsf{Unblind}')$ *is a partially blind signature scheme,* $\{f_s\}_{s\in\{0,1\}^*}$ *is a family of PRFs, and* $(P, V, \mathbf{O})$ *is an oSE NIZK for the languages* L*, then the scheme* Γ *is an unlinkable ad-hoc survey scheme that is secure against malicious users.*

Before proving Theorem 6, let us remark that by instantiating all the above-mentioned primitives with generic constructions (see Theorem 3, 4), Theorem 6 directly implies the following result.

Theorem 7 *Assume the existence of enhanced trapdoor permutations. Then there exists a correct, unlinkable ad-hoc voting scheme that is secure against malicious users.*

Let us now turn to proving Theorem 6.
Proof: (of Theorem 6). We start by showing Unlinkability.

Unlinkability Towards proving unlinkability, we consider a sequence of hybrid experiments:

– Hybrid 1 = EXEC^0.

– Hybrid 2 is identical to Hybrid 1 except that all NIZK proofs are simulated (More precisely, we consider an attacker A' and witness selector W' for simulation-extractability experiment of the NIZK: A' incorporates A and emulates the experiment EXEC^0 for A except that whenever the experiment dictates that A receives an NIZK, A' request an NIZK for the same statement and tag from its prover oracle and let W' be the machine that outputs the witnesses used for the proofs in EXEC^0.) It directly follows by the correctness of the simulation property of $(P, V, \mathbf{O})$, and the fact that in EXEC^0 we only invoke P on true statements for which P is provided a valid witness (this follows from the fact that we always run $\mathsf{Authorized}$ before invoking P which ensures that P has a valid witness) that the output of Hybrid 2 is indistinguishable from Hybrid 1.

– Hybrid 3 is identical to Hybrid 2 except that in the execution of $\mathsf{RegUser}^U$, honest users blind 0^n (as opposed to the seed s), but still use the seed s later on in the experiment). Note that in Hybird 3 honest users never use the signature received from the RA—the only way the message received by the RA is used is to determine

whether the signature is accepting or not (this follows from the fact that all NIZK proofs are simulated in Hybrid 3, and in the real execution the signature is only used as a witness to the NIZKs). Consequently, it follows from the weak partial blinding property, and a straight-forward hybrid argument (over the number of registration queries), that the output of Hybrid 3 is indistinguishable from Hybrid 2.

- Hybrid 4 is identical to Hybrid 3 except that in the execution of SubmitSurvey, we use a truly random function (which can be efficiently simulated using lazy evaluation) instead of using a PRF. Consequently, it follows from the pseudorandomness property of the PRF, and a straight-forward hybrid argument (over the number of different users and thus PRFs), that the output of Hybrid 4 is indistinguishable from Hybrid 3.

- Hybrid 5 is identical to Hybrid 4, except that b (of EXEC^b) is set to 1. First, note that conditioned on $\mathsf{Authorized}(\mathsf{vk}_{\mathsf{SA}}, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, id_\beta) = \mathsf{fail}$ for some β or A querying its $\mathsf{SubmitSurvey}(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, \cdot, \mathsf{cred}_{id_\beta})$ oracle, the outputs of the Hybrid 5 and Hybrid 4 are identical (since it will be fail). Second, conditioned on $\mathsf{Authorized}(\mathsf{vk}_{\mathsf{SA}}, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, id_\beta) = \mathsf{accept}$ for both $\beta \in \{0,1\}$ and A never queried its $\mathsf{SubmitSurvey}(1^n, \mathsf{vid}, \mathsf{vk}_{\mathsf{vid}}, \cdot, \mathsf{cred}_{id_\beta})$ oracle, it follows by independence property of the random function that the outputs of Hybrid 5 and Hybrid 4 are identically distributed (since the only difference between the two experiments is the order in which we evaluate two different random functions on vid, without ever evaluating these function of vid any other times).

- Hybrid 6 is identical to Hybrid 3, except that we set b is set to 1 (i.e., we, switch back using a PRF); indistinguishability of the outputs of Hybrid 6 and 5 follows by the same argument as indistinguishability between Hybrid 4 and 3.

- Hybrid 7 is identical to Hybrid 2, except that we set b is set to 1 (i.e., we, switch back to request a blinding of s); indistinguishability of the outputs of Hybrid 7 and 6 follows by the same argument as indistinguishability between Hybrid 3 and 2.

- Hybrid 8 = EXEC^1 (i.e., we, switch back to using real NIZK proofs); indistinguishability of the outputs of Hybrid 8 and 9 follows by the same argument as indistinguishability between Hybrid 1 and 2.

We have thus concluded that EXEC^0 and EXEC^1 are indistinguishable, and thus unlinkability follows.

We are now ready to prove security.

Security Consider some nuPPT that breaks security with probability $1/p(n)$ for infinitely many n for some polynomial p. As above, consider a hybrid experiment, which

is identical to the real experiment except that all NIZK proofs received by A are simulated. It directly follows by the correctness of the simulation property of $(P, V, \mathbb{O})$, and the fact that in Hybrid 1 we only invoke P on true statements for which P is provided a valid witness, that A succeeds in breaking security with probability, say, $1/2p(n)$ for infinitely many n. Additionally, note that in this hybrid experiment, when implementing the $\mathsf{SubmitSurvey}'$ oracle for A, we never actually have to generate a (blind) signature on the seed s_{id} of some honest party id—this signture was only used as a witness to NIZK proofs generated in survey submission but all those proofs are now simulated without the witnesses, so whether these signatures are generated or not does not affect the experiment.

Now, note that whenever A is successful in breaking security, then by condition 5, it holds that tags for the NIZK proofs in all submissions output by A are different from the tags of all NIZKs in $\mathsf{SubmitSurvey}'$ queries; thus, by the correctness property of the online extractor X for the NIZK, with overwhelming probability, for every submission output by A, X will extract out the identity, seed, and appropriate signatures for that submission, given only the oracle queries made by A (and their answers). By the unforgeability of $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$, each such identity belongs to L (or else we have obtained a signature on a new message). By the unforgeability of $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}', \mathsf{Blind}', \mathsf{Unblind}')$, each such identity also belongs to $L_{corrupted}$ (as mentioned above, in the hybrid experiment we consider, we no longer requests signatures for identities in L_{honest} and thus the only signatures generated in the experiment are for identities in $L_{corrupted}$). Thus, the number of distinct identities is at most $|L \cap L_{corrupter}|$. Additionally, since the RA only agree to sign a single (blinded) seed s_{id} per identity id, it follows (again) from the unforgeability of $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}', \mathsf{Blind}', \mathsf{Unblind}')$ that the number of distinct seeds s_{id} is $|L \cap L_{corrupted}|$. But since the PRF is deterministic (once the seed has been fixed), it follows that that there can be at most $|L \cap L_{honest}|$ different token numbers in submissions output by A, which implies that A does not break security. □

5 Concrete Instantiation

In this section, we provide concrete instantiations of a PRF, a partially blind signature scheme, and an NIZK proof system for language L that are all based on a group that supports a bilinear pairing. We briefly describe our setting.

Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be groups of prime order q. A *bilinear map* is an efficient mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, which is both: (*bilinear*) for all $g \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and $a, b \leftarrow \mathbb{Z}_q$, $e(g^a, g_2^b) = e(g, g_2)^{ab}$; and (*non-degenerate*) if g generates $\mathbb{G}_1$ and g_2 generates $\mathbb{G}_2$, then $e(g, g_2)$ generates $\mathbb{G}_T$. Let $\mathcal{G}(1^\lambda)$ be an efficient deterministic (but potentially non-uniform[11]) algorithm that on input 1^λ produces the description $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g, g_2, e, H)$

[11] Constructing an elliptic curve of a given size typically requires random coins. We assume these coins are given non-uniformly to make the construction algorithm deterministic for analysis (in practice, we simply

of a prime $q = \Theta(2^\lambda)$, descriptions of cyclic groups $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_T$ of order q along with a generator g for $\mathbb{G}_1$, a generator g_2 for $\mathbb{G}_2$, a bilinear map e, and a description of a hash function modeled as a random oracle H that maps $\{0,1\}^* \to \mathbb{Z}_q$. In our case, $\mathcal{G}$ produces the description of a Barreto-Naehrig pairing-friendly curve [36].

5.1 Our Assumptions

Our schemes rely on the Pedersen commitment scheme, a signature scheme implicitly used in the Boneh-Boyen identity-based encryption scheme, and the Dodis-Yampolskiy pseudo-random function, all described in Figure 1. We assume that these schemes are secure with respect to our group generator $\mathcal{G}$. The Pedersen commitment scheme is secure assuming that discrete logarithm is hard in the groups produced by $\mathcal{G}$ [37]. The later two schemes have been well-studied and versions have been reduced to the (sub-exponential hardness of) Decisional Bilinear Diffie-Hellman assumption and the n-Decisional Diffie-Hellman Inversion Assumption respectively. See [38, Section 4.3], [10], and [39] for details.

We will later refer to the following assumptions.

Assumption 1 *When instantiated with group generator $\mathcal{G}(\lambda)$, the Pedersen commitment scheme is a perfectly-hiding and computationally-binding commitment scheme.*

Assumption 2 *When instantiated with group generator $\mathcal{G}(\lambda)$, the Boneh-Boyen scheme in Fig. 1 is an unforgeable signature scheme for λ-bit messages.*

Assumption 3 *When instantiated with group generator $\mathcal{G}(\lambda)$, the Dodis-Yampolskiy function in Fig. 1 is a secure pseudo-random function that maps λ-bit strings to elements of $\mathbb{G}_T$.*

Theorem 8 (Security of the Survey System) *Under Assumptions 1, 2 and 3, the concrete instantiation in section 5.4 is a correct (Def. 6), unlinkable (Def. 7) ad-hoc survey scheme that is secure against malicious users (Def. 8) in the random oracle model.*

Proof: By Assumption 1 and 3, we have a commitment scheme and a PRF scheme. By Assumption 2 and Theorem 10 in Section 5.2.1, we show how to instantiate a secure partially-blind signature scheme. In Theorem 13 in Section 5.3, we instantiate an online extractable NIZK proof system for language L. The concrete implementation in Section 5.4 is an instantiation of the abstract implementation from Section 4.3 with these primitives. Thus, the security of our system follows from the combination of these three primitives with Theorem 3 from Section 4.1 concerning simulation extraction and finally the abstract implementation Theorem 6. □

use well-studied curves selected by experts). We require this because our modular security analysis assumes that the four cryptographic primitives are run independently. In practice, however, all four primitives must run using the same group parameters. We can ensure this by assuming that our group generator $\mathcal{G}$ is deterministic. Technically, this makes all of our cryptographic primitives non-uniform constructions (since $\mathcal{G}$ is now possibly non-uniform), but we ignore this slight mismatch as it has no impact.

(1) Commitment Scheme

Commitment Key: Receiver publishes generators $g, h \in \mathbb{G}$.

Commit(m) $S \longrightarrow R$ Sender chooses random $s \in \mathbb{Z}_q$, and sends $\alpha = g^m h^s$ to R.

Open(α)**:** $S \longrightarrow R$ Sender sends (m, s) to R. Receiver checks $\alpha \stackrel{?}{=} g^m h^s$.

(2) Dodis-Yampolskiy PRF F_y that maps $\mathbb{Z}_q \to \mathbb{G}_T$ where $y \in \mathbb{Z}_q$.

Setup: A group $\mathbb{G}_T$ of prime order q with generator $e(g, g_2)$ and PRF seed $y \in \mathbb{Z}_q$.

$F_y(m)$**:** The user computes $F_y(m) = e(g, g_2)^{1/(y+m)}$ for any m such that $(m + y) \neq 0 \mod q$.

(3) BB Signature Scheme

$\mathsf{Gen}(1^n)$: Sample the secret key $\mathsf{sk} \leftarrow \alpha \in \mathbb{Z}_q$. Sample random group generators $u, v, h \in \mathbb{G}_1$ and compute $U = e(g, g_2)^\alpha$. The verification key is $\mathsf{vk} \leftarrow (u, v, h, U)$.

$\mathsf{Sign}(\mathsf{sk}, m_0, m_1)$**:** Choose $r \in \mathbb{Z}_q$ randomly and compute

$$\sigma_1 \leftarrow g^\alpha (u^{m_0} v^{m_1} h)^r, \quad \sigma_2 \leftarrow g_2^r, \quad \sigma_3 \leftarrow g^r$$

and output $(\sigma_1, \sigma_2, \sigma_3)$ as the signature.

$\mathsf{Ver}(\mathsf{vk}, m_0, m_1, \sigma_1, \sigma_2)$**:** Accept if $e(\sigma_1, g_2) \stackrel{?}{=} U \cdot e(u^{m_0} v^{m_1} h, \sigma_2) \wedge e(\sigma_3, g_2) = e(g, \sigma_2)$

Figure 1: A commitment, and PRF family, and signature scheme

5.2 Primitives

The common input for all protocols is the output $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g, g_2, e)$ of the group generating algorithm and a description of a hash function modeled as a random oracle H that maps $\{0,1\}^* \to \mathbb{Z}_q$.

5.2.1 Instantiation of Partially-Blind Signature Scheme

Our instantiation of a partially-bling signature scheme roughly follows the abstract construction presented in Definition 14, with the commitment scheme instantiated by the Pedersen scheme, and the signature scheme instantiated with the Boneh-Boyen signa-

ture scheme. The difference is that instead of computing a signature on the commitment, the signer manages to directly compute a *blinded* signature on the message. In order to show the unforgeability of our scheme, we add an online-extractable NIZK proof to the commitment message to certify that the commitment is well-formed. The language for this proof is:

$$L_{\text{blind}} = PoK\left\{(s_{id}, d) : \alpha = v^{s_{id}} g^{d}\right\}$$

which corresponds to a standard Schnorr [40] zero-knowledge proof of knowledge of integers s_{id}, d such that $\alpha = v^{s_{id}} g^{d}$ holds" where α, v, g are elements of a group $\mathbb{G}$.

$\mathsf{Gen}'(1^n)$: Sample the secret key $\mathsf{sk} \leftarrow \alpha \in \mathbb{Z}_q$. Sample random group generators $u, v, g, h \in \mathbb{G}_1$ and compute $U = e(g, g_2)^{\alpha}$. The verification key is $\mathsf{vk} \leftarrow (u, v, g, g_2, h, U)$.

$\mathsf{Blind}'_r(\mathsf{vk}, s)$**:** If g is not a generator, output $\perp$. Else, use random tape r to choose $d \in \mathbb{Z}_q$, compute $\alpha = v^s g^d$. Compute an online extractable (NI)zero-knowledge proof of knowledge for $(s, d) \in L_{\text{blind}}$:

$$\pi = PoK\left\{(s, d) : \alpha = v^{s} g^{d}\right\}$$

Output (α, π).

$\mathsf{Sign}'(\mathsf{sk}, (m, \alpha))$**:** Choose $w \in \mathbb{Z}_q$ randomly and compute

$$\sigma_1 \leftarrow g^{\alpha}(u^m \alpha h)^w, \quad \sigma_2 \leftarrow g_2^w \qquad \sigma_3 \leftarrow g^w$$

and output $(\sigma_1, \sigma_2, \sigma_3)$ as the signature.

$\mathsf{Unblind}'_r(\mathsf{vk}, (\sigma_1, \sigma_2, \sigma_3))$**:** Use tape r to fix $d \in \mathbb{Z}_q$ and output the pair $(\sigma_1 / \sigma_3^d, \sigma_2)$.

$\mathsf{Ver}'(\mathsf{vk}, (m, s), (\sigma_1, \sigma_2))$**:** Accept if $e(\sigma_1, g_2) \stackrel{?}{=} U \cdot e(u^m v^s h, \sigma_2)$.

Figure 2: A semi-blind signature scheme based on the Boneh-Boyen signature scheme

Theorem 9 ([40]) *There exists a 3-round honest-verifier special-sound zero-knowledge protocol for L_{blind}.*

The result of applying Theorem 2 to the protocol from Theorem 9 is an online-extractable NIZKPoK for language L_{blind}.

Theorem 10 *The partially-blind signature scheme in Fig. 2 is weakly-blind.*

Proof: Consider some malicious signer S^*, and some strings $z, \mathsf{vk}, m, s_0, s_1$. Through a sequence of hybrids, we show that the experiments $\mathsf{EXEC}^0(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)$ and $\mathsf{EXEC}^1(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)$ are indistinguishable.

We first define an oracle $O(\mathsf{vk}, (m, \alpha), (x_1, x_2, x_3))$ to output 1 if and only if $e(x_1, g_2) \stackrel{?}{=} U \cdot e(u^m \alpha h, x_2)$ and $e(g, x_2) \stackrel{?}{=} e(x_3, g_2)$. Consider the following sequence of hybrid experiments.

Hybrid $H_1(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)$ This experiment is the same as EXEC_0 except that line (3) of the experiment is replaced by computing $o \leftarrow O(\mathsf{vk}, (m, \alpha), x)$. By the following lemma which states that the answer o computed in this way always matches the answer computed in EXEC_0, these two experiments are identical.

Lemma 11 $O(\mathsf{vk}, (m, \alpha), (x_1, x_2, x_3)) = \mathsf{Ver}'(\mathsf{vk}, (m, s), \mathsf{Unblind}'_r(\mathsf{vk}, (x_1, x_2, x_3)))$

Proof: Note that

$$e(u^m \alpha h, x_2) = e(u^m (v^s g^d) h, x_2) = e(u^m v^s h, x_2) e(g^d, x_2)$$

and if $e(g, x_2) \stackrel{?}{=} e(x_3, g_2)$ holds, then $e(g^d, x_2) = e(x_3^d, g_2)$. Expanding, we have:

$$\begin{aligned}
\mathsf{Ver}'(\mathsf{vk}, (m, s), \mathsf{Unblind}'_r(\mathsf{vk}, (x_1, x_2, x_3))) &= \mathsf{Ver}'(\mathsf{vk}, (m, s), (x_1 / x_3^d, x_2)) \\
&= 1 \text{ iff } e(x_1 / x_3^d, g_2) \stackrel{?}{=} U \cdot e(u^m v^s h, x_2) \\
&= 1 \text{ iff } e(x_1, g_2) \stackrel{?}{=} U \cdot e(u^m v^s h, x_2) e(x_3^d, g_2) \\
&= 1 \text{ iff } e(x_1, g_2) \stackrel{?}{=} U \cdot e(u^m v^s h, x_2) e(g^d, x_3) \\
&\qquad \wedge e(g, x_2) \stackrel{?}{=} e(x_3, g_2) \\
&= O(\mathsf{vk}, (m, \alpha), (x_1, x_2, x_3))
\end{aligned}$$

where the second to last line follows from the substitution above. □

Since vk, (m, α) and (x_1, x_2, x_3) are all publicly available in the EXEC experiment, it follows that line (3) of the experiment EXEC^b can be computed directly.

Hybrid $H_2(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)$ Same as H_1 except that the NIZK proof in the Blind operation is generated using the simulator S instead of the prover algorithm P. By the special HVZK property of the online extractable NIZK system (Def. 10), the output of H_2 is indistinguishable from Hybrid H_1.

Hybrid $H_3(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)$ Instead of sending the message $v^{s_0} g^d$ as the result of the Blind operation, send $v^{s_1} g^d$. Hybrid H_3 is identically distributed to H_2 by the perfect hiding property of the Pedersen commitment (Theorem 1).

Hybrid $H_4(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)$ Replace the simulator for the NIZK proof with the real prover algorithm run using witness s_1, d. Hybrids H_3 and H_4 are indistinguishable based on the zero-knowledge property of the NIZK system as discussed in hybrid H_2.

$\mathsf{EXEC}^1(1^n, S^*, z, \mathsf{vk}, m, s_0, s_1)$ Replace the use of oracle O with the instructions from EXEC_1. Since all steps are either identically distributed or computationally indistinguishable, it follows that experiments $\mathsf{EXEC}_0(\cdots)$ and $\mathsf{EXEC}_1(\cdots)$ are indistinguishable. □

Theorem 12 *Assuming the security of the Boneh-Boyen signature scheme, the scheme in Fig. 2 is unforgeable.*

Proof: Suppose that there exists some nuPPT A, and a polynomial p such that for infinitely many $n \in \mathbb{N}$, A makes ℓ signature queries of the form (m_i, α_i, π_i) but manages to output $\ell + 1$ valid message-signature pairs $(m_i, s_i), (\sigma_1, \sigma_2)$ (for unique messages s_i) with probability $p(n)$.

We use A to construct an adversary A' that breaks the unforgeability of the Boneh-Boyen signature scheme with probability that is polynomially related to $p(n)$. Adversary A' works as follows: it receives public key vk for the Boneh-Boyen scheme and begins a simulation of the partial-blind unforgeability game for A by running $A(\mathsf{vk})$. A' intercepts and records every query to the oracle H into the set Q. When A submits a partial-blind signature query of the form (m, α, π), then A' first checks the proof π. If the proof π verifies, then A' runs the extraction function $(s, d) \leftarrow X(\pi, Q)$ using the set of random oracle queries made by A up to this point of the execution. If X fails, then A' aborts. A' submits the message pair (m, s) to the signature-query oracle for the Boneh-Boyen scheme and receives a signature $(\sigma_1, \sigma_2, \sigma_3)$. A' then computes the pair $(\sigma_1 \cdot (\sigma_3)^d, \sigma_2)$ and returns it to the adversary A. At the end of this simulation, after asking ℓ queries, the adversary A outputs $\ell + 1$ pairs $\{(m_i, s_i), (\sigma_{1,i}, \sigma_{2,i})\}$ where each s_i is unique and A' outputs the same.

We now analyze the success probability of A'. First, conditioned on the event that X never fails during extraction, we argue that the simulation for A is identically distributed to the partial-blind game. Differences may only occur when responding to A's signature queries (m, α, π) (observe that the vk fed as input to A in the first step, and the answers from the oracle H are identically distributed to the partial-blind game). In the partial-blind game, A receives a triple of elements $\sigma_1 = g^x(u^m \alpha h)^r$, $\sigma_2 = g_2^r, \sigma_3 = g_1^r)$ where r is chosen randomly. In our simulation, A receives a triple (x_1, x_2, x_3) where $x_1 = g^x(u^m v^s g^d h)^r$, $x_2 = g_2^r$, $x_3 = g^r$ subject to the constraint that $\alpha = v^s g^d$, and thus the simulation is perfect.

Finally, we argue that by the online extraction property (Def 13) of the NIZK system and the union bound over the polynomial number of queries issued by A, it follows that the extractor X fails with at most negligible probability. Thus, the success probability of A' in outputting forgeries for the Boneh-Boyen scheme is at least $(1 - \epsilon(n))p(n)$ for

a negligible function ϵ. Combining this with Assumption 2 which state that the Boneh-Boyen scheme is unforgeable, we conclude that $p(n)$ must also be negligible. $\square$

5.3 Instantiation of NIZK proof system for L

We now describe a Σ-protocol for L and then apply Theorem 3 to produce the required online-extractable NIZK for language L. A statement in L consists of the tuple $(g, g_2, \mathsf{vid}, C, \mathsf{vk}_{\mathsf{RA}}, \mathsf{vk}_{\mathsf{SA}})$ where $\mathsf{vk}_{\mathsf{RA}} = (u, v, h, e(g, g_2)^x)$ and $\mathsf{vk}_{\mathsf{SA}} = (u_v, v_v, h_v, e(g, g_2)^y)$. The witness for an instance is the tuple $(s_{id}, id, c, r, \sigma, \sigma_{\mathsf{vid}_{id}})$ such that $\sigma = (\sigma_1, \sigma_2)$ forms a Boneh-Boyen signature on the values (id, s_{id}), $\sigma_{\mathsf{vid}_{id}} = (\sigma_{\mathsf{vid}_{id},1}, \sigma_{\mathsf{vid}_{id},2})$ forms a Boneh-Boyen signature on (vid, id), and $C = F_{s_{id}}(\mathsf{vid}) = e(g, g_2)^{1/(s_{id}+\mathsf{vid})}$ where F is the Dodis-Yampolskiy PRF.

In the first step of the proof for L, the prover re-randomizes $(\sigma, \sigma_{\mathsf{vid}_{id}})$ by choosing random $d_1, d_2 \in \mathbb{Z}_q$ and computes

$$\begin{aligned}(s_1 = \sigma_1 \cdot (u^{id} v^{s_{id}} h)^{d_1} \quad , \quad s_2 = \sigma_2 \cdot g_2^{d_1}) \\ (s_3 = \sigma_{\mathsf{vid}_{id},1} \cdot (u_v^{\mathsf{vid}} v_v^{id} h)^{d_2} \quad , \quad s_4 = \sigma_{\mathsf{vid}_{id},2} \cdot g_2^{d_2}).\end{aligned}$$

The values s_2, s_4 are sent to the Verifier, and the problem reduces to proving the statement: (a) (s_1, s_2) form a Boneh-Boyen signature on the values (id, s_{id}), (b) (s_3, s_4) form a Boneh-Boyen signature on (vid, id), and (c) $C = F_{s_{id}}(\mathsf{vid})$ as follows:

$$PoK \left\{ \begin{array}{l} (id, s_{id}, s_1, s_3) : \\ \quad E^x \cdot e(h, s_2) = e(s_1, g_2) \cdot e(u^{id} v^{s_{id}}, s_2)^{-1} \wedge \\ \quad E^y \cdot e(u_v^{\mathsf{vid}} h_v, s_4) = e(s_3, g_2) \cdot e(v_v^{id}, s_4)^{-1} \wedge \\ \quad E \cdot C^{-\mathsf{vid}} = C^{s_{id}} \end{array} \right\}$$

where $E = e(g, g_2)$ as follows:

1. $P_2 \rightarrow V_2$ Prover picks random $b_1, b_2 \in \mathbb{Z}_q$ and $J_1, J_2 \in G$ and computes

$$\begin{aligned} E_1 &\leftarrow e(J_1, g_2) \cdot e(u^{b_1} v^{b_2}, s_2)^{-1} \\ E_2 &\leftarrow e(J_2, g_2) \cdot e(v_v^{b_2}, s_4)^{-1} \\ E_3 &\leftarrow C^{b_2} \end{aligned}$$

2. $P_2 \leftarrow V_2$ Verifier picks a random $c \in \mathbb{Z}_q$.

3. $P_2 \rightarrow V_2$ Prover computes a response

$$\begin{aligned} z_1 &\leftarrow b_1 + c \cdot id & z_2 &\leftarrow b_2 + c \cdot s_{id} \\ z_3 &\leftarrow s_1^c \cdot J_1 & z_4 &\leftarrow s_3^c \cdot J_2 \end{aligned}$$

4. Verifier checks the following:

$$E_1 \cdot e(g, g_2)^{xc} \cdot e(h, s_2)^c = e(z_3, g_2) \cdot e(u^{z_1} v^{z_2}, s_2)^{-1}$$
$$E_2 \cdot e(g, g_2)^{yc} \cdot e(u_v^{\mathsf{vid}} h_v, s_4)^c = e(z_4, g_2) \cdot e(v_v^{z_1}, s_4)^{-1}$$
$$E_3 \cdot e(g, g_2)^c \cdot C^{-c(\mathsf{vid})} = C^{z_2}$$

Although not specifically written, the Verifier must perform standard checks that each element of the proof is indeed a member of the appropriate group.

Theorem 13 *(P_2, V_2) is an honest-verifier special-sound zero-knowledge protocol for L.*

Proof: The completeness of the protocol is standard. First we show honest-verifier zero-knowledge. On input an instance and a random challenge c, the simulator first chooses a random $z_1, z_2 \in \mathbb{Z}_q$ and random $z_3, z_4 \in G$ and computes

$$E_1 = \frac{e(z_3, g_2) \cdot e(u^{z_1} v^{z_2}, s_2)^{-1}}{e(g, g_2)^{cx} \cdot e(h, s_2)^c}$$
$$E_2 = \frac{e(z_4, g_2) \cdot e(v_v^{z_1}, s_4)^{-1}}{e(g, g_2)^{cy} \cdot e(u_v^{\mathsf{vid}} h_v, s_4)^c}$$
$$E_3 = \frac{C^{z_2}}{e(g, g_2)^c \cdot C^{-c(\mathsf{vid})}}$$

and outputs $(E_1, E_2, E_3), c, (z_1, z_2, z_3, z_4)$ as the transcript. By inspection, it follows that the distribution of transcripts is perfectly identical to a transcript from a successful protocol execution.

We now show that the protocol is special-sound. Consider the elements of two verifying transcripts $(s_2, s_4, E_1, E_2, E_3)$, $c, (z_1, z_2, z_3, z_4)$ and $(s_2, s_4, E_1, E_2, E_3), c', (z_1', z_2', z_3', z_4')$ that share the same first message but where $c \neq c'$. We use these group elements to solve for the witness values (id, s_{id}, s_1, s_3). Since both transcripts verify, it follows that

$$E_1 \cdot e(g, g_2)^{xc} \cdot e(h, s_2)^c = e(z_3, g_2) \cdot e(u^{z_1} v^{z_2}, s_2)^{-1} \quad (1)$$
$$E_1 \cdot e(g, g_2)^{xc'} \cdot e(h, s_2)^{c'} = e(z_3', g_2) \cdot e(u^{z_1'} v^{z_2'}, s_2)^{-1} \quad (2)$$
$$E_2 \cdot e(g, g_2)^{yc} \cdot e(u_v^{\mathsf{vid}} h_v, s_4)^c = e(z_4, g_2) \cdot e(v_v^{z_1}, s_4)^{-1} \quad (3)$$
$$E_2 \cdot e(g, g_2)^{yc'} \cdot e(u_v^{\mathsf{vid}} h_v, s_4)^{c'} = e(z_4', g_2) \cdot e(v_v^{z_1'}, s_4)^{-1} \quad (4)$$
$$E_3 \cdot e(g, g_2)^c \cdot C^{-c(\mathsf{vid})} = C^{z_2} \quad (5)$$
$$E_3 \cdot e(g, g_2)^{c'} \cdot C^{-c'(\mathsf{vid})} = C^{z_2'} \quad (6)$$

Dividing equation (1) by (2), equation (3) by (4), and equation (5) by (6), we have:

$$e(g, g_2)^{x(c-c')} \cdot e(h, s_2)^{(c-c')} = e(z_3/z'_3, g_2) \cdot e(u^{z_1-z'_1} v^{z_2-z'_2}, s_2)^{-1} \tag{7}$$

$$e(g, g_2)^{y(c-c')} \cdot e(u_v^{\mathsf{vid}} h_v, s_4)^{(c-c')} = e(z_4/z'_4, g_2) \cdot e(v_v^{z_1-z'_1}, s_4)^{-1} \tag{8}$$

$$e(g, g_2)^{(c-c')} \cdot C^{-(c-c')(\mathsf{vid})} = C^{(z_2-z'_2)} \tag{9}$$

Finally, rearranging, and raising each side to $1/(c-c')$, we have

$$\begin{aligned} e(g, g_2)^{x} \cdot e(u^{(z_1-z'_1)/(c-c')} v^{(z_2-z'_2)/(c-c')} h, s_2) &= e((z_3/z'_3)^{1/(c-c')}, g_2) \\ e(g, g_2)^{y} \cdot e(u_v^{\mathsf{vid}} v_v^{(z_1-z'_1)/(c-c')} h_v, s_4) &= e((z_4/z'_4)^{1/(c-c')}, g_2) \\ e(g, g_2) &= C^{(z_2-z'_2)/(c-c')+\mathsf{vid}} \end{aligned}$$

Thus, by inspection, it follows that $\left(\left(\frac{z_3}{z'_3}\right)^{1/(c-c')}, s_2\right)$ verifies as a signature on the tuple (id, s_{id}) where

$$\begin{aligned} id &= \frac{z_1 - z'_1}{c - c'} \\ s_{id} &= \frac{z_2 - z'_2}{c - c'} \end{aligned}$$

and $\left(s_3 = \left(\frac{z_4}{z'_4}\right)^{1/(c-c')}, s_4\right)$ verifies as a signature on the tuple (id, vid), and C is $F_{s_{id}}(\mathsf{vid})$. Thus, we have shown that two transcripts of the proper form can be used to extract witnesses for the theorem statement. □

Corollary 14 *In the random oracle model, there exists an online extractable NIZK for L.*

Proof: Follows from Thm. 13 and Thm. 3. □

5.4 Concrete scheme

For convenience, we combine all building blocks to describe the full scheme.

GenRA **and** GenSA The RA picks random group elements $u, v, h \in \mathbb{G}_1$ and a secret element $x \in \mathbb{Z}_q$. The RA's public key $\mathsf{vk}_{\mathsf{RA}} = (u, v, h, e(g, g_2)^x)$ and $\mathsf{sk}_{\mathsf{RA}} = x$. (RA will be signing m_1 as the id with u and m_2 as the user's secret seed with v.)

The SA picks random group elements $u_v, v_v, h_v \in \mathbb{G}_1$ and a secret element $y \in \mathbb{Z}_q$. The SA's public key $\mathsf{SA}_{\mathsf{vk}} = (u_v, v_v, h_v, e(g, g_2)^y)$ and $\mathsf{SA}_{\mathsf{sk}} = y$. ($m_1$ will be the vid and m_2 will be the user id of a participant authorized to submit in the survey.)

The RegUser **protocol** A user registers by invoking the following protocol with the RA:

Common: $\mathsf{RA}_{vk} = (u, v, h, e(g, g_2)^x)$

RA Secret Key: x

User identity: id

User and RA establish a mutually authenticated secure communication channel.

$U \to \mathsf{RA}$ The user chooses a random PRF seed $s_{id} \in \mathbb{Z}_q$ and a random $d \in \mathbb{Z}_q$, computes $\alpha = v^{s_{id}} g^d$, and sends (id, α) to RA.

The user also gives a (NI)zero-knowledge proof of knowledge for $(s_{id}, d) \in L_1$ using the Σ-protocol for L_1 described above:

$$PoK\left\{(s_{id}, d) : \alpha = v^{s_{id}} g^d\right\}$$

$U \to \mathsf{RA}$ User picks a random b_1, b_2 and sends $\gamma = v^{b_1} g^{b_2}$ to RA.

$U \to \mathsf{RA}$ User generates a random challenge $c \in \mathbb{Z}_q$ by using the random-oracle H and the tag 0^n as $c = H(g, \mathsf{RA}_{vk}, id, \alpha, \gamma, 0^n)$

$U \to \mathsf{RA}$ User computes $z_1 = b_1 + c s_{id}$, $z_2 = b_2 + cd$ and sends (z_1, z_2) to RA.

RA verifies $v^{z_1} g^{z_2} \stackrel{?}{=} \alpha^c \gamma$.

$U \longleftarrow \mathsf{RA}$ RA checks that the identity id has not been registered before. RA chooses $r \in \mathbb{Z}_q$ randomly, computes the signature tuple $\sigma_1 \leftarrow g^x (u^{id} \alpha h)^r, \sigma_2 \leftarrow g_2^r$ and sends R the signature $\sigma_{id} = (\sigma_1, \sigma_2)$.

U User verifies the signature by checking that

$$e(\sigma_1, g_2) = e(g, g_2)^x \cdot e(u^{id} v^{s_{id}} g^d h, \sigma_2).$$

If this verifies, the user removes the commitment randomness by computing $\sigma_1' = \sigma_1 / \sigma_2^d$ and stores the secret credential $(id, s_{id}, \sigma_{id} = (\sigma_1, \sigma_2))$.

Survey Registration To create a survey:

SA Input: $\mathsf{SA}_{vk} = (u_v, v_v, h_v, e(g, g_2)^y)$, $\mathsf{SA}_{\mathsf{sk}} = y$, $\mathsf{vid} \in \mathbb{Z}_q$

List of identies: L

SA For each $id \in L$, the SA computes the following:

Pick a random $r \in \mathbb{Z}_q$ and compute

$$\sigma^{\mathsf{vid}_{id}} = (g^y (u_v^{\mathsf{vid}} v_v^{id} h)^r, g_2^r)$$

Publish the list $L_{\mathsf{vid}} = (\mathsf{vid}, \{id_i, \sigma^{\mathsf{vid}_{id}}\}_{i \in L})$

Authorized**:** Anyone can verify that a user with identity *id* is authorized to submit in survey vid by finding the corresponding signature $\sigma^{\mathsf{vid}_{id}} = (\sigma_1, \sigma_2)$ in L_{vid} and then checking that

$$e(\sigma_1, g_2) \stackrel{?}{=} e(g, g_2)^y \cdot e(u_v^{\mathsf{vid}} v_v^{id_i} h, \sigma_2).$$

Submission To submit a survey:

Common Input: List L_{vid}, the public keys $\mathsf{SA}_{vk} = (u_v, v_v, h_v, e(g, g_2)^y)$, and $\mathsf{RA}_{vk} = (u, v, h, e(g, g_2)^x)$

User Secrets: *id*, submission *m*, credential (σ_{id}, s_{id})

The user aborts if the user has already participated in an survey with vid or $\mathsf{vid} = s_{id}$. The user and SA establish a secure connection in which SA is authenticated, but the user is anonymous.

U The user identifies the tuple $(\mathsf{vid}, id_i, \sigma^{(i)})$ in L_{vid} in which $id_i = id$. The user computes $F_{s_{id}}(\mathsf{vid}) = C \leftarrow e(g, g_2)^{1/(s_{id} + \mathsf{vid})}$.

$U \rightarrow \mathsf{SA}$ User sends $(\mathsf{vid}, C, m, s_2, s_4)$ and an NIZKPOK of the statement (id, s_{id}, s_1, s_3) in *L* with tag $1||\mathsf{vid}||m$ to the SA:

$$PoK \left\{ \begin{array}{l} (id, s_{id}, s_1, s_3) : \\ \quad e(g, g_2)^x e(h, s_2) = e(s_1, g_2) e(u^{id} v^{s_{id}}, s_2)^{-1} \wedge \\ \quad e(g, g_2)^y e(u_v^{\mathsf{vid}} h_v, s_4) = e(s_3, g_2) e(v_v^{id}, s_4)^{-1} \wedge \\ \quad e(g, g_2) \cdot C^{-\mathsf{vid}} = C^{s_{id}} \end{array} \right\}$$

SA : If the proof verifies, record the submission (C, m) replacing any prior occurrence of $(C, \cdot)$.

Acknowledgments

All opinions expressed and implied in this work are solely those of the authors and do not represent or reflect the views of their respective universities.

References

[1] S. Staff, "Security breach leaves 45,000 at risk of identity theft," 2009. [Online]. Available: http://thetruthwillrise.wordpress.com/2009/06/25/security-breach-leaves-45000-at-risk-of-identity-theft/ (Cited on 1.)

[2] P. C. A. for Submission., "Observations on the Course Review System at the University of Virginia," 2013. (Cited on 1.)

[3] M. Riley, "U.s. agencies said to swap data with thousands of firms," 2013. [Online]. Available: http://www.bloomberg.com/news/2013-06-14/u-s-agencies-said-to-swap-data-with-thousands-of-firms.html (Cited on 1.)

[4] D. Chaum and T. P. Pedersen, "Wallet databases with observers," in *CRYPTO*, vol. 740, 1992, pp. 89–105. (Cited on 1 and 6.)

[5] K. Sako and J. Kilian, "Receipt-free mix-type voting scheme – a practical solution to the implementation of a voting booth," in *Eurocrypt 1995*, 1995. (Cited on 1 and 6.)

[6] A. Neff, "A verifiable secret shuffle and its application to e-voting," in *CCS 2001*, 2001. (Cited on 1 and 6.)

[7] J. Benaloh, "Simple verifiable elections," in *EVT 2006*, 2006. (Cited on 1 and 6.)

[8] B. Adida, "Helios: Web-based open-audit voting," in *USENIX 2008*, 2008. (Cited on 1 and 6.)

[9] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *CRYPTO*, 2004, pp. 56–72. (Cited on 4 and 6.)

[10] J. Camenisch, S. Hohenberger, M. Kohlweiss, A. Lysyanskaya, and M. Meyerovich, "How to win the clonewars: Efficient periodic n-times anonymous authentication," in *ACM CCS '06*, 2006, pp. 201–210. (Cited on 4, 6, and 30.)

[11] A. Sahai, "Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security," in *FOCS'99*, 1999, pp. 543–553. (Cited on 5 and 16.)

[12] R. Pass and A. Rosen, "Concurrent non-malleable commitments," *SIAM Journal of Computing*, 2008. (Cited on 5, 16, and 17.)

[13] ——, "New and improved constructions of non-malleable cryptographic protocols," *SIAM Journal of Computing*, 2008. (Cited on 5, 16, and 17.)

[14] D. Chaum and E. van Heyst, "Group signatures," in *EUROCRYPT '91*, 1991, pp. 257–265. (Cited on 6.)

[15] M. Bellare, D. Micciancio, and B. Warinschi, "Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions," in *EUROCRYPT*, 2003, pp. 614–629. (Cited on 6.)

[16] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *CRYPTO '04*, 2004, pp. 45–55. (Cited on 6.)

[17] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *ASIACRYPT '01*, 2001, pp. 552–565. (Cited on 6.)

[18] M. Bellare, H. Shi, and C. Zhang, "Foundations of group signatures: The case of dynamic groups," in *CT-RSA*, 2005, pp. 136–153. (Cited on 6.)

[19] D. Chaum, "Security without identification: Transaction systems to make big brother obsolete." *Communications of the ACM*, vol. 28(10), pp. 1030–1044, October 1985. (Cited on 6.)

[20] J. Camenisch and A. Lysyanskaya, "Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation," in *EUROCRYPT '01*, vol. 2045, 2001, pp. 93–118. (Cited on 6.)

[21] ——, "A signature scheme with efficient protocols," in *SCN*, 2002, pp. 268–289. (Cited on 6.)

[22] J. Camenisch, S. Hohenberger, and A. Lysyanskaya, "Compact e-cash," in *EUROCRYPT '05*, 2005, pp. 302–321. (Cited on 6.)

[23] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28(2), pp. 270–299, 1984. (Cited on 8.)

[24] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM J. Computing*, vol. 17(2), pp. 281–308, 1988. (Cited on 8.)

[25] M. Naor and M. Yung, "Universal one-way hash functions and their cryptographic applications," in *STOC '89*, 1989, pp. 33–43. (Cited on 8.)

[26] J. Rompel, "One-way functions are necessary and sufficient for secure signatures," 1990. (Cited on 8.)

[27] O. Goldreich, S. Goldwasser, and S. Micali, "How to Construct Random Functions," *Journal of the ACM*, vol. 33, no. 4, pp. 792–807, 1986. (Cited on 8.)

[28] J. Håstad, R. Impagliazzo, L. Levin, and M. Luby, "A pseudorandom generator from any one-way function," *SIAM Journal on Computing*, vol. 28, pp. 12–24, 1999. (Cited on 8 and 9.)

[29] M. Naor, "Bit commitment using pseudorandomness," *Journal of Cryptology*, vol. 4, no. 2, pp. 151–158, 1991. (Cited on 9.)

[30] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in *FOCS '01*, 2000, see updated version at Cryptology ePrint Archive: Report 2000/067. (Cited on 16 and 19.)

[31] A. D. Santis, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Robust non-interactive zero knowledge," *SIAM Journal on Computing*, vol. 20, pp. 1084–1118, 2001. (Cited on 16 and 19.)

[32] B. Barak, R. Canetti, J. B. Nielsen, and R. Pass, "Universally composable protocols with relaxed set-up assumptions," in *45th Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society, 2004, pp. 186–195. (Cited on 19.)

[33] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *CRYPTO*, 1994, pp. 174–187. (Cited on 19.)

[34] R. Pass, "On deniability in the common reference string and random oracle model," in *CRYPTO 2003*, 2003, pp. 316–337. (Cited on 19 and 20.)

[35] J. B. Nielsen, "Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case," in *CRYPTO 2002*, 2002, pp. 111–126. (Cited on 20.)

[36] P. S. L. M. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," in *Selected areas in cryptography*, 2006, pp. 319–331. (Cited on 30.)

[37] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *CRYPTO*, 1991, pp. 129–140. (Cited on 30.)

[38] Y. Dodis and A. Yampolskiy, "A Verifiable Random Function with Short Proofs and Keys," in *PKC '05*, vol. 3386 of LNCS, 2005, pp. 416–431. (Cited on 30.)

[39] D. Boneh and X. Boyen, "Efficient selective-ID secure Identity-Based Encryption without random oracles." in *EUROCRYPT '04*, 2004, pp. 223–238. (Cited on 30.)

[40] C.-P. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptography*, vol. 4, pp. 161–174, 1991. (Cited on 32.)