PRIMO: Privacy-Preserving Merifiable Crowd Counting for Protests

Daniel Bosk

Abstract—**The problem of participation estimation is a fundamental democratic issue in the context of physical protests or demonstrations. There, the number of participants reflects the strength of the support or at least interest of the public society towards the topic of the protest. Current methods for crowd counting have wide margins for error in addition to being generally quite privacy-invasive. Moreover, the organizers of the protest may be tempted to cheat by inflating the number of participants as much as possible whereas it is in the government's interest to do the opposite if the protest is directed against it. In this paper, we propose a novel approach to this problem by combining location proofs with some properties of electronic voting to provide verifiable yet privacy-preserving participation counts. Since the parties have incentives to cheat, transparency is a fundamental requirement that we achieve through a decentralized scheme providing both verifiability and privacy. More specifically, the proposed scheme is distributed, does not depend on any authority and it must scale to handle large protests (i.e., millions of participants). In our scheme, each participant creates a participation proof with the collaboration of other participants, called witnesses, in a privacy-preserving manner (i.e., without requiring to reveal their own identity or that of witnesses). The blockchain is used to commit and timestamp the participation proofs, which essentially consist of location proofs in which a privacy-preserving distance-bounding protocol is used by a participant to prove their proximity to the witnesses. Our scheme provides verifiability, which means that each participant can verify that their participation has been included in the final count— i.e., individual verifiability, to prevent the regime in place from dropping data — and that anyone can verify the correctness of the result — i.e., universal and eligibility verifiability, to prevent any malicious actor (e.g., Sybil attack) from affecting the result.** Sonja ►*more scientific results?*'◄

I. Introduction

Context. Consider the following illustrative scenario. Alice is an activist that organizes a protest in some location(s) against the current government, represented by Grace. On the one hand, Alice wants to estimate the number of participants to prove a certain support for her cause. On the other hand, Grace is an autocratic regime leader who opposes Alice's cause. Bob, Carol and many others participate in the protest for Alice's cause against Grace. However, they might only be willing to do so in a manner that does not leave any digital traces because of the fear of retribution against them from Grace's regime. To realize this objective a reliable yet privacy-preserving crowd-counting mechanism is needed, which to the best of our knowledge is a problem that has not yet been entirely solved.

Historically, there are many examples of protests in which the count estimated by police and that by the organizers differ significantly, sometimes in the order of hundreds of thousands. Even without foul play, the difference is quite natural as both parties have different objectives and metrics. More precisely, the organizers want to count everyone who participated while the police want to estimate the count at the peak of participation, due to crowd control [1]. Among the numerous recent examples in which it is difficult to establish the actual number of participants, there are the demonstrations against the president in South Korea [1], Trump's inauguration attendance [2], the 2017 Women's March in the US [3], the demonstrations against the change of constitution in Venezuela [4] or for the independence of Catalonia [5].

Existing methods for crowd-counting vary significantly in terms of approaches (we will review them in details in Section IV). However, most of them lack precision (i.e., they have large error margins) and they can only give an estimate for a particular snapshot in time (e.g., at the peak of the event). In particular, the existing methods cannot reliably estimate the cumulative participation count — at least not without counting some persons multiple times, which in turn increases the error of the estimation. In addition, they also lack verifiability.

Finally, one important observation about crowd-counting that has not been adequately addressed in the design of current crowd-counting solutions is that it is actually an adversarial setting. Indeed, going back to our example, Alice the activist has an incentive to increase the tallied number of participants, whereas Grace (and possibly other entities) have an incentive to decrease it. In this case, we are left with two options: either we trust Alice or Grace (or another third party), or we have to be able to verify their claims ourselves. In this paper, our main objective is to provide a scheme preventing both Alice and Grace from cheating (i.e., their claim should be verifiable). To solve this issue, we need a verifiable participation count that can resist Sybil attacks while still preserving the participants' privacy to the extent possible given their physical presence at the protest. More precisely, our solution will not be able to prevent an observer (physically present or looking at photos or videos) from recognizing a particular individual at a protest. Nonetheless, we can ensure that none of the digital traces generated by a participant in our scheme can be associated with their identity and that any witnessing is also unlinkable.

Contributions. In this paper, our main contribution is to propose PRIVO, a privacy-preserving scheme providing a way to securely estimate crowd count in protests. This verification system does not incur any additional risk to protesters' privacy beyond the ones related to physically participating in the protest in the first place.

Our second contribution is an adaptation of the Schnorr protocol [6] for distance bounding (DB) [7], but which is different from that of Brands and Chaum [7]. Our adaptation fulfils all the requirements for modern DB protocols, including distance hijacking (DH) [8] (which was an attack against Brands-Chaum). More importantly, it is public key (i.e., han-

dles malicious, impersonating verifiers), fulfils terrorist fraud (TF) [9] resistance and provides a proof of knowledge (PK) for discrete logarithms. The latter allows us to replace the use of the plain Schnorr protocol in the zero-knowledge proof of knowledge (ZKPK) used in anonymous credentials and thus we are, to the best of our knowledge, the first to provide DB anonymous credentials.

Limitations and assumptions. Some of the assumptions that are required for implementing our proposition are not yet realized (briefly, use of smartphones, unique identities, distance bounding - see Section II for more detailed descriptions). In particular, not all (or even many) protesters, especially in countries with oppressive regimes, currently possess smartphones. In addition, digital certificates signed by a central authority are not yet widely available and if we rely on government-issued credentials, we cannot prevent the government from performing Sybil attacks — since it can issue arbitrarily many *valid* credentials. In this case, it does not make sense to verify any pro-government protests in which Grace's authoritarian regime wants to demonstrate its wide support, e.g. [4], [10]. Finally, achieving distance-bounding protocols on smartphones is currently not feasible within a meaningful range of distance needed for our scenario. They lack the required hardware to do the distance bounding fast enough.

However, we believe that some assumptions might become sufficiently realistic in the near future, due to the smartphone penetration being on the rise and the increased digitalization and government-issued credentials available in some countries already now (e.g., Estonia, Germany and Sweden). In addition, the distance-bounding chips[1] currently available can already enable proofs of proximity for any range of up to 200 meters. One of the short-term objectives is to be able to integrate them in phones or smartcards in the near-future, and even phones with off-the-shelf hardware running in RFID-emulation mode have shown promising results [11].

PRIVO achieves privacy, but not receipt freeness. This means that there is a way for Grace to efficiently verify that Alice has participated. However, this requires Grace to be able to get a copy of Alice's secret key so that she can simply redo the (deterministic) computations on the same inputs (the probability of collisions being negligible).

Outline. The paper is organized as follows. First, we define and formalize the notion of protest and discuss its relationship with electronic voting (Section II). We then present the desired security properties for such a system (Section III) and subsequently discuss related work in terms of these properties (Section IV). We give the relevant background on the building blocks of our solution, including our proposal for distance-bounding anonymous credentials that cope with a malicious (impersonating) verifier (Section V). PRIVO, the protocol constructed from the building blocks, is presented next (Section VI) and we analyze its security and privacy (Section VII) as well as performance and implementation issues (Section VIII). We conclude with a discussion.

[1]https://www.3db-access.com

II. Defining and Formalizing Protest and Crowd Estimation

Defining the concept of protest. To be able to estimate the participation count for a protest, we first need to define this concept and what should be counted. Let us start by considering some examples. During the demonstrations against the South Korean president in Seoul "[t]he rallies stretch[ed] from midday to late night — some people stay[ed] for several hours, others just several minutes" [1]. These rallies were all in the same location in the capital and repeated every weekend for the duration of a few weeks. The Women's Marches [3], on the other hand, occurred in parallel in many locations. We also have the Venezuelan demonstrations in which "anti-government demonstrators have staged daily protests across Venezuela" [12] while "pro-government workers sang and danced as they staged a rival march to show their support for the president's controversial plan to rewrite the constitution" [4]. Judging from these examples, the minimal common part is the cause, but there is also a location (or area) that varies over time.

For the rest of the paper, we will refer to the organizer as Alice. Assume that the objective of Alice is to count everyone who participated at any time and in any of the locations [1][2].

Formally, we define a protest as an event that is uniquely identified by its cause (*cid* below), its time interval (t) and its location (area l). More specifically, we will use the following definition.

Definition 1 (Protest)**.** *A* subprotest *is a tuple* (cid, t, l) *in which* $cid \in \mathbb{Z}_{2^\lambda}$ *is the identifier of the cause of the protest,* $t \subseteq \mathbb{R}$ *is a time period and* $l \subseteq \mathbb{R}^2$ *is the location (topological connectedness is not necessary). A* protest *is a set of subprotests sharing the same cid.*

The protests described in the previous examples can be captured using this definition by splitting them up into subprotests. Each subprotest will then be captured by our definition and to estimate the total participation to the protest we can just sum up the estimates obtained. Similarly for marches, the marching path can be divided into subprotests with locations (or areas) that slightly overlap.

Formalizing the notion of crowd estimation. Each participant who wants to be counted must submit a *participation proof*. The proof must be associated with the protest (i.e., its identifier *cid*), time and location must coincide with any of its subprotests.

We use *witnesses* to associate the proof to the location by creating a *proof share*. A witness is only allowed to create one proof share per protester to avoid the risk of inflation. The participation proof and its proof shares are stored in a set, as there should be only unique proof shares.

Definition 2 (Participation-proof share)**.** *A* participation-proof share $s = (cid, t, l, pid, wid)$ *is a tuple where:* cid, t, l *are as in Definition 1;* pid *is a protester's pseudonym for the protest identified by* cid*; and* wid *is a witness's pseudonym*

[2]Note that the objective does not necessarily align with the definition of the police whose interest could be to determine the maximum crowd at any point in time, to deploy enough personnel for crowd-control.

for a protester with pseudonym pid. We say that s is part of a subprotest $p = (cid', t', l')$ iff $cid = cid', t \subseteq t', l \subseteq l'$ and denote this by $s \sqsubseteq p$. We let S be the set of all proof shares.

A subset of the proof shares forms a participation proof for a protester.

Definition 3 (Participation proof)**.** *A* participation proof *of protester with pseudonym pid participating in a protest $P = \{(cid, t_i, l_i)\}_i$ is the set*

$$\pi_{pid,P} = \{s = (cid, t, l, pid, wid) \in S \mid \exists p \in P \colon s \sqsubseteq p\}$$

of all proof shares with the same protester and protest identifiers, witnessing time interval t and location l is within those of the protest. We denote by Π the set of all proofs.

We can now define the participation count as follows.

Definition 4 (Participation count)**.** *We define the* participation count *of a protest P (as in Definition 3) as the cardinality $|\Pi_P^{\varsigma,\theta}|$ of the set of participation proofs*

$$\Pi_P^{\varsigma,\theta} = \{\pi_{i,P} \in \Pi \mid \varsigma(\pi_{i,j}) \geq \theta\}$$

for some strength function $\varsigma \colon \mathcal{P}(S) \to \mathbb{R}_+$ and a threshold θ.

The strength function ς can be used to regulate trust in witnesses. One example would be for ς to return the number of unique witnesses and thus let θ to be the threshold of the number of required witnesses. Another example would be to return the number of proof shares issued by trusted witnesses and set $\theta = 1$ to require at least one proof share issued by a trusted witness.

System model and physical assumptions. Throughout this text, when we refer to a participant, say Alice, we actually mean an agent which can perform cryptographic operations and communicate with other local devices on Alice's behalf. We assume that every participant has a digital certificate signed by some logically centralized certificate authority. E.g., that we can use the cryptographic keys of any national electronic identity system, identity card or passport. We need this to prevent Sybil attacks [13].

In practical terms, participants witness each other's participation using their smartphones (or similar devices) running the protocol described in Section VI and uploading their testimony (i.e., proof shares) to a blockchain after the protest. During the protest the devices are computationally limited by their batteries and need local connectivity to each other but no connection to any global network such as the Internet is necessary. Before and after the protest, we assume that the devices have global connectivity, i.e., Internet connections, and are not computationally limited by any battery.

Since the cellular network can be shut down to keep protesters from accessing the Internet, making phone calls or texting, we require a different means of communications between protesters. This can be accomplished by Bluetooth or WiFi, as demonstrated by Briar [14] and FireChat [15], two examples of apps for communication during protests via wireless mesh networking. An alternative, although originally designed to work within 5G cellular networks, is device-to-device communication (D2D) [16]. Specifically, the outband and autonomous version D2D fits our scenario thanks to using unlicensed spectrum and working without cellular coverage.

III. Security Requirements

In our setting, we have two potential adversaries: Alice and Grace. Alice is a participant to the protest and thus she might be also interested in manipulating the system to increase the count (e.g., she might have an incentive to do so as an activist). In contrast, the aim of Grace, the totalitarian dictator, is to decrease the count but also additionally to deanonymize some of the participants in order to arrest them or forcefully convince them to change their minds. In the following two subsections, we will define the desired security properties.

A. Verifiability

We note that, in general terms, protesting is very similar to petitions, which in turn are similar to voting: all three situations correspond to many individuals expressing their opinion. These opinions can be sensitive (e.g., be a cause for discrimination or persecution), hence we desire to have similar properties of verification and privacy for verifying a protest as there are for voting.

Voting has (generally) three desirable requirements for verifiability [17].

Eligibility: anyone can verify that each vote cast is legitimate.
Universal verifiability: anyone can verify that the result is according to the cast votes.
Individual verifiability: every voter can verify that their vote is included in the result.

We translate the votes into *participation proofs*. Universal and individual verifiability remain the same: anyone can verify the participation count by counting the proofs. The eligibility requirement is slightly different: for protests the eligibility requirement must include temporal and spatial eligibility (i.e., each participation proof satisfies some temporal and spatial relation to the protest). In essence, the proof must bind the person to the time and location of the protest.

To define these properties more formally in the context of protest, we desire three verifiability requirements, among which eligibility can be further broken up into four subproperties:

V1. *Eligibility*: anyone can verify that each participation proof provides temporal and spatial eligibility and that only one participation proof is counted per individual.
V1.1. *Temporal eligibility*: prove that the proof was created after the start of the protest and before the end of the protest.
V1.2. *Spatial eligibility*: prove that the proof is spatially related to the physical location or journey of the protest.
V1.3. *One-proof-per-person*: prove that no individual can be counted more than once for a particular protest.
V1.4. *Designated event*: prove that the proof is designated for the protest.
V2. *Universal verifiability*: anyone can verify that the result obtained match the submitted participation proofs.

V3. *Individual verifiability*: each participant can verify that their participation proof is included in the global count.

B. Privacy

In addition to the verification requirements, we also need to define the privacy properties. In voting protocols, there are also three levels of privacy [17]:

Vote privacy: the voting does not reveal any individual vote.
Receipt freeness: the voting system does not provide any data that can be used as a proof of how the voter voted.
Coercion resistance: a voter cannot cooperate with a coercer to prove the vote was cast in any particular way.

Delaune, Kremer, and Ryan [17] showed that coercion resistance implies receipt freeness, which in turn implies vote privacy. Coercion resistance is probably not possible to achieve for protests: e.g., Grace can simply physically bring Alice to a protest against her will.

In essence, receipt freeness means that upon completing the protocol, Grace cannot link Alice to Alice's participation proof — even if she were to compromise Alice's device or Alice collaborates with Grace. As we will see, if Grace gets Alice's device, she can re-do the computations on the same inputs. Since, in our case, the algorithms are deterministic and collisions are negligible she will get the same result and can thus conclude that Alice is indeed the one she is looking for. Thus we cannot provide receipt freeness. (See Section IX for further discussion.)

This leaves us with what should correspond to vote privacy. We need unlinkability between Alice's long-term identity and Alice's participation proof. Given a participation proof, Grace will not be able to tell if it belongs to Alice or Bob. Furthermore, if Grace has managed to link one proof to Alice's, she should not be able to link another proof as a consequence. Hence, we can summarize our needed properties as follows:

P1. Participants must be unlinkable to their proofs.
P2. Participants (or proofs for the same participant) must be unlinkable between different protests.

Thus Alice can always argue to be part of the protest in Grace's favour instead of any protest against Grace, which, in fact, is a slight increase in privacy over ordinary protests.

IV. Related work

A. Crowd counting methods

The seemingly most commonly used method for counting crowds at protests is *Jacobs's method*, cf. [1], [2], [18]–[20]. It is a manual method devised in the 1960s relying on aerial photos of the event. The verifier divides the protest venue into regions and then estimates the density of the crowd in the different regions and finally sums them up to get an estimate of the global count. Clearly this method is prone to errors as it is based on estimates. It provides universal verifiability (requirement V2) since anyone can redo the counting using the same photos. However, it is difficult to achieve individual verifiability (requirement V3) using this method since it is hard to verify that oneself is indeed in any of the photos[3]. We can argue that it also achieves eligibility (requirement V1) if all the photos are taken at the same point in time with no overlap, since then no one can be counted twice. Unfortunately, we can only get the peak participation with this method. When it comes to spatial and temporal eligibility, we have to rely on pictures (e.g., by determining from the photos themselves when and where the they were taken). However, this is prone to cheating — and subject to a variety of misinformation techniques — and the best we can do is to employ forensic methods to try to detect any modifications or other attempts at fraud.

From a privacy perspective, all techniques based on photo or video material are as privacy preserving as the protesters make themselves. There is a risk that participants are recognizable from the material if they do not wear masks.

Among more recent methods, there is an application called CrowdSize [21] that can estimate the size of a crowd based on a selected space. Once the user has specified the zone, he selects one of three pre-set density estimates: light, medium or dense. This is similar to the method usually used by police (e.g., for the protests in Seoul): "[p]olice presume[d] that, when sitting, six people would fill a space of 3.3 square meters [...] The same area would hold nine or 10 people when standing" [1]. This method is not verifiable unless the user takes a picture of the crowd, in which case it inherits the verifiability properties of the previous method.

In the computer vision community, there is a body of work on estimating the number of persons in a picture (e.g., the work of Zhang, Li, Wang, *et al.* [22]). This class of methods requires photos or video surveillance of the protest location during the entire protest and are thus highly privacy-invasive. They are generally based on machine learning, and thus also require a training dataset to work. In the work by Zhang, Li, Wang, *et al.* [22], they actually train and evaluate their algorithm on different scenes, which might make this method easier to use for protesting. Universal verifiability (requirement V2) can be provided with this method, since someone can always recount the participants using the recorded video material. Some degree of individual verifiability could also be provided (requirement V3), if Alice can recognize her own face in the video, but this might still be difficult. Note also that while automatic face recognition can help verifiability, it has a high impact on privacy. Also, it is difficult (if not impossible) to not count people twice, thus we cannot argue for eligibility (requirement V1). In theory, this class of methods will provide an upper bound, whereas our method outputs a lower bound. However, it is still difficult to capture the entire location on surveillance video, which diminishes the argument for an upper bound in practice. The spatial and temporal verifiability properties are reduced to those of the video, which should be similar as in our discussion about the verifiability of photos (i.e., using forensic methods).

Another problem for all the above methods is exemplified by the demonstrations in Seoul: "[t]he demonstrators not only gather in open space, but also small alleys and between buildings" [1]. In this situation it is very difficult to faithfully capture the situation. Even if the photo material is taken from different angles, we will have the additional problem of not

[3]It is of course possible with very detailed photos, but not currently realistic to do with aerial photos.

counting people twice.

During the protests in Seoul [1], one physical analytics company tried to estimate the number of participants using their technology. They scanned for MAC addresses emitted from the Wi-Fi of participants' smartphones. However, this method required many assumptions:

> The company presumed that about half of smartphone users usually leave their Wi-Fi feature on and the other half switch it off, based on a separate survey on smartphone usage. It also assumed that about 20% of the smartphone signals were repetition from the same device. [1]

This method cannot provide any verifiability, as we must trust the company in doing the measurements correctly and to be honest about when and where they did them. As MAC address randomization get wider adoption, this will be even more difficult, although some tracking of smartphones could still be possible [23].

A better proposal would be to use *IMSI catchers* (or the real cell towers of the mobile network) to count unique phones at a location. However, there are several problems with this approach. First, it will be difficult to register only the participants' phones, as many bystanders will also be counted. It is also difficult to differentiate between the protesters and the counter-protesters. Second, since the phone has a unique identifier, participants might be uncomfortable to be registered in association with the event and might thus turn the device off, even though, at least with 5G, the IMSI will not be transmitted in plaintext anymore. Finally, there will be limited verifiability, as the data recorder must be trusted to record all data in the relation to the protest.

In general for all of the above methods, the more a crowd spreads out, the more difficult it will be to determine its size. In particular, one of the challenge is determining whether people near the event's perimeter are participants or simply bystanders [24]. These methods also have difficulty capturing the actual attendance of an event (i.e., the cumulative participation, not just the count at a snapshot around the peak).

One of the most closely related work to our scheme is CrowdCount [25], which is a web service that lets Alice create an event such that anyone can submit his location to register that he is in Alice's event. This method has the benefit of counting everyone who has declared his presence, not just the count at the snapshot of the photos. However, there is no verification as the service must be trusted to behave honestly, but even then, nothing prevents Bob from submitting twice (violating eligibility, requirement V1). Another downside is that the service also requires an Internet connection during the event to register as a participant. This makes it prone to some form of denial-of-service attack. For instance, if Alice has organized a protest against Grace's regime, who shuts down the cellular network or Internet backbone as a means to censor the protest.

Another related approach based on devices is UrbanCount [26], which uses epidemic spreading of crowd-size estimates by device-to-device communication to count crowds in dense urban environments with high node-mobility and churn. There is, however, no consideration of a potentially adverserial setting and thus no verification or checks on eligibility. DiVote [27], a prior work by the same authors for polling in dense areas, avoids double counting, but again, only work for honest participants.

B. Location proofs

Some location-based service (LBS) only grant access to resources to users located at a particular location, thus raising the issue of verifying the position claimed by a particular user. In most current schemes, the location of a user/device is determined by the device itself (e.g., through GPS) and forwarded to the LBS provider. One of the main drawback of this approach is that a user can cheat by having his device transmitting a false location. Therefore, it is possible for a user to be inappropriately granted access to a particular resource while being thousands of kilometers away.

To counter this threat, an LBS should ask the requesting device to formally prove that it really is at the claimed location. This notion can be formalized through the concept of *location proof*. In a nutshell, a location proof is a digital certificate attesting that someone was at a particular location at a specific moment in time. A location proof architecture is a system by which users can obtain location proofs from neighboring witnesses (e.g., trusted access points or other users) that can later be shown to verifiers who can check the validity of a particular proof. Most of the existing approaches to location proofs require the prover and the witnesses to disclose their identities, thus raising many privacy issues such as the possibility of tracing the movements of users of the location proof architecture. Seb ►*to do add the corresponding reference*◄ However, some location proofs systems, such as PRivacy-preserving lOcation-Proof System (PROPS) [28], exists that provides strong privacy guarantees along with the possibility of verifying the claim of the location. PRIVO share some similarities with PROPS, although their objective is quite different as we aim at verifying a global property of the population (i.e., crowd estimation) in contrast to checking the location claim made by a user, which is an individual property.

V. Building blocks

In this section, we will briefly review the primitives that form the building blocks of PRIVO.

A. Collision-resistant hash functions

One primitive that we need is a collision-resistant hash function, H.

H can be instantiated with any algorithm from the SHA-2 or SHA-3 families.

B. Zero-knowledge proofs of knowledge

We will use the notation introduced by Camenisch and Stadler [29]:

$$\mathrm{PK}\{(\alpha,\beta,\gamma) : y = g^{\alpha}h^{\beta} \wedge y' = \hat{g}^{\gamma}\} \qquad (1)$$

which means that we prove knowledge of α, β, γ ensuring that y, y' are of the form $y = g^\alpha h^\beta$ and $y' = \hat{g}^\gamma$, respectively. Greek letters are known only to the prover and for which the prover wishes to prove knowledge, all other letters are known by the verifier.

When a proof of knowledge is turned into a signature using the Fiat-Shamir heuristic [30], we will denote it as

$$\sigma \leftarrow \mathrm{SPK}\{(\alpha, \beta, \gamma) : y = g^\alpha h^\beta \wedge y' = \hat{g}^\gamma\}(m),$$

which yields a signature σ on m, ensuring that the issuer knew α, β, γ such that $y = g^\alpha h^\beta$ and $y' = \hat{g}^\gamma$.

C. Anonymous credentials

We need an anonymous credential system, AC, which provides the following algorithms and properties.

AC must provide a commitment scheme, AC.Commit, and algorithms such that the prover can convince a verifier that he knows the commitment, which means that:

$$\mathrm{PK}\{(k, o) : c = \mathsf{AC.Commit}(k, o)\}.$$

We will require that the commitment scheme is *perfectly hiding* and computationally binding, rather than the other way around[4].

AC must also provide a (blindable) signature scheme with protocols to get a signature on a committed value (AC.GetSig $\leftrightarrow$ AC.IssueSig) and to prove knowledge of a signature on a committed value (AC.ProveSig $\leftrightarrow$ AC.VerifySig).

The prover commits to a value k with commitment $c \leftarrow \mathsf{AC.Commit}(k, o)$ and opening o. Afterwards, he uses $\sigma \leftarrow \mathsf{AC.GetSig} \leftrightarrow \mathsf{AC.IssueSig}$ to obtain a signature $\sigma = \mathsf{AC.Sign}_{sk}(k)$ on the value k, in which sk is the signing key of the signer.

At a later point, the prover wants to prove to a verifier that he knows k and a signature σ on k made by the owner of pk (corresponding to sk), i.e., without revealing k nor σ. The prover and verifier will run the protocol AC.ProveSig $\leftrightarrow$ AC.VerifySig to prove the following:

$$\mathrm{PK}\{(k) : \sigma' = \mathsf{AC.BlindSig}(\mathsf{AC.Sign}_{sk}(k))\}.$$

Finally we need a pseudo-random function, AC.PRF, such that there exists a protocol AC.ProvePRF $\leftrightarrow$ AC.VerifyPRF which implements the following PK:

$$\mathrm{PK}\{(k) : y = \mathsf{AC.PRF}_k(x)\}.$$

This means that the prover can convince the verifier that $y = \mathsf{AC.PRF}_k(x)$ without revealing k.

Instantiations. The AC scheme can be instantiated using the Pedersen commitment [31] for AC.Commit, CL-signatures [32] (or adapted CL-signatures as in [33]) for AC.Sign, AC.BlindSig, AC.GetSig, AC.IssueSig and the verifiable random function by Dodis and Yampolskiy [34] as AC.PRF. These are used together in e.g., Anon-Pass [33] and by Camenisch, Hohenberger, Kohlweiss, *et al.* [35] to form unclonable anonymous credentials for subscriptions. We will, however, modify the ZKPK protocols slightly, by making them distance bounding.

[4]We are more concerned with long-term privacy, and thus we are looking for information-theoretic hiding.

D. Distance bounding

DB protocols were first suggested by Brands and Chaum [7] to prevent relay attacks in contactless communications in which the adversary forwards a communication between a prover and a possibly far-away verifier to authenticate. These attacks cannot be prevented by cryptographic tools as they are independent of the semantics of the messages exchanged, so mechanisms ensuring the physical proximity between a verifier and a prover must be used. DB protocols precisely enable the verifier to estimate an upper bound on his distance to the prover by measuring the time-of-flight of short challenge-response messages (or rounds) exchanged during time-critical phases. Time critical phases are complemented by slow phases during which the time is not taking into account. At the end of a DB protocol, the verifier should be able to determine if the prover is legitimate and in his vicinity.

The main security requirements of DB protocols can be summarized as follows:

- Mafia fraud (MF) [36]: the adversary illegitimately authenticates, possibly using a far-away honest prover.
- terrorist fraud (TF) [9]: a legitimate but malicious prover helps a close-to-the-verifier accomplice to authenticate.
- distance fraud (DF) [7]: a legitimate but malicious prover wants to fool the verifier on the distance between them.
- distance hijacking (DH) [8]: similar to DF, sometimes using the presence of an honest prover close to the verifier.
- impersonation fraud (IF) [37]: the adversary plays against a simplified version of the protocol without any distance estimation.

There are two lines of attempts at formalizing the above properties: one by Boureanu, Mitrokotsa, and Vaudenay [38] and another by Dürholz, Fischlin, Kasper, *et al.* [39].

The majority of the existing DB protocols are symmetric and thus requires an honest verifier. Indeed, in this context it does not make sense to protect against the verifier as he can easily impersonate the prover as he has a knowledge of his secret key. There has been less work done in the domain of asymmetric (or public-key) DB protocols. Unfortunately, our setting requires a public-key DB protocol with a *malicious verifier* who will try to *impersonate the prover*. The verifier might also try to track the provers and map their identities to their actions, thus we also require privacy. This leads to the requirement of a DB ZKPK, or simply distance-bounding proof-of-knowledge (DBPK), with resistance to all the frauds mentioned above as well as against this verifier who will attempt to impersonate the prover. More specifically, we must combine such a DB with anonymous credentials (which we do in Section V-E). The anonymous credential system will affect what properties we require of the DBPK, (e.g., that it must be a public-key encryption for discrete logarithms).

E. Distance bounding anonymous credentials

There is a protocol which provide DBPK with all the desired properties from above, ProProx [40]. (Note that [40] uses the abbreviation PoPoK, for Proof of Proximity of Knowledge, instead of DBPK.) However, ProProx provides a DBPK

protocol for quadratic residues (i.e., protocols of the form DBPK$\{(\alpha) : a = \alpha^2\}$) while in our context, we need a DBPK protocol for discrete logarithms (i.e., DBPK$\{(\alpha) : a = g^\alpha\}$). To realize this, we will now introduce such a protocol that will allow us to do DB anonymous credentials.

Reattempting a distance bounding Schnorr protocol. In the original DB paper by Brands and Chaum [7], they show how to design a distance-bounding protocol based on the Schnorr identification scheme [6]. That DB Schnorr protocol was shown to be prone to distance hijacking [8]. (It was not secure against terrorist fraud either.) We now propose another way to turn the Schnorr protocol into a DB protocol which is secure against MF, DF, DH, TF and an impersonating verifier (i.e., it is public-key).

We present the protocol in Fig. 1. The difference from the original Schnorr protocol is that the prover commits to one random value but the verifier sends two challenges. The verifier presents to the prover which commitment and challenge to use during the DB phase.

The intuition behind the protocol security is as follows. To achieve malicious-verifier zero-knowledge we must choose k logarithmically in the security parameter λ and repeat the protocol until the knowledge error is small enough. This will actually also decrease the success probability of an MF adversary in the DB step. Indeed, the MF adversary does not know which challenge the verifier will use, thus there is a $1/2$ probability that the MF adversary will guess it correctly. The prover will only provide one of the responses: even if the adversary re-runs the protocol with the same challenges, the randomly chosen ρ_0, ρ_1 will have changed — which thus yields another R_b in the end of the protocol. A sequential repetition of protocol will decrease this probability.

The DB phase protects against distance fraud. A DF prover must wait for the challenge bit b_i before responding with r_i.

The DB phase also ensures that an MF adversary will fail the DB phase for at least one round. The verifier will send $\lceil \log_2 q \rceil + l < 2 \lceil \log_2 q \rceil$ challenges. Thus the prover reveals all the bits of either s_0 or s_1 but only l bits of the other. Thus, the adversary can buffer l bits of both s_0, s_1 by requesting them from the prover. However at this point the adversary must wait for b_{l+1} from the verifier and then relay that challenge to the prover to receive the correct r_i — this relay will be detected.

The protocol is TF resistant, if the malicious prover gives both responses to the adversary, the adversary can compute the secret key. This follows from the fact that it is a ZKPK, i.e., there is an extractor that can extract the secret.

The protocol is also secure against distance hijacking due to the fact that it is the authenticating bit string that is used during the DB phase, not the challenge bit string as in Brands-Chaum.

TODO DONE

F. Time-stamping service – blockchains

We need a time-stamping service, TS, with algorithms TS.Get, TS.Stamp and TS.Time such that

- $\rho \leftarrow$ TS.Get yields a value ρ at time t, ρ is difficult to guess before time t and TS.Time$(\rho) = t$;
- $\pi \leftarrow$ TS.Stamp(x) yields a value π at time t such that TS.Verify$(x, \pi) \rightarrow 1$ and TS.Time$(\pi) = t$.

With these building blocks we can ensure that a message m was created within the time interval $[t_0, t_1]$. After time t_0, request $\rho_{t_0} \leftarrow$ TS.Get. Before time t_1, submit $h \leftarrow \mathsf{H}(m, \rho_{t_0})$ to the time-stamping service to get $\pi_{t_1} \leftarrow$ TS.Stamp(h).

Now we can use $(\rho_{t_0}, m, \pi_{t_1})$ to prove that m was created within the time interval $[t_0, t_1]$. The verifier computes $h' \leftarrow \mathsf{H}(m, \rho_{t_0})$ and checks whether TS.Verify$(h', \pi_{t_1}) = 1$ and TS.Time$(\rho_{t_0}) = t_0 \wedge$ TS.Time$(\pi_{t_1}) = t_1$.

We can instantiate TS using a blockchain, e.g., OmniLedger [41]. The TS.Stamp(x) algorithm will simply include x in the blockchain and return the identifier of the block into which x was included. TS.Get will return the hash of the current head of the chain. This value is difficult to predict since it depends on all submitted transactions and usually some randomness (e.g., nonces and miners' secrets). The blockchain must also be continuously extended, such as in Bitcoin [42], in which a new block appears approximately every 10 minutes. Having a dedicated blockchain for a protest will not do, and thus it is better to use a blockchain which is used for other ~~things~~ services such as proposed in Aspen [37].

There are some additional properties that we need from TS, which are implied by blockchains. First, we need *immutability*, to ensure that once we commit something (through TS.Stamp) it will remain there. We will use this to ensure verifiability as the data must remain to be verifiable. Second, once we have committed some data, we do not need to keep the data itself, but we can store some confirmation value instead. For instance using blockchains, we can store the hash of the block in which our data was committed, while this block remains we are sure that our data remains committed — even if we no longer remember what our data looked like.

VI. PRIVO: A PROTOCOL FOR PROTEST OUTCOME VERIFICATION

We now present PRIVO, a protocol for securely and PRIvately Verifying PROtest Outcomes. The protocol consists of four phases: the setup, join, participation and submission phases. We have several roles. We have a certificate authority (CA) who is responsible for a one-to-one mapping between a person's identity and a cryptographic key. We have the protesters who want to participate in a given protest. A protester can assume three different roles: (1) We have the *organizer* who will simply write a manifesto for the protest and spread to others. (2) A *participant* participates in the protest (and asks witnesses to vouch for his presence). (3) A *witness* provides proofs to participants that state the participant was indeed participating and the proofs are verifiable by third parties. In general, there is one organizer and every protester will act as both participant and witness.

The setup and registration phases are the same as in AnonPass [33]. We have simply adapted their description for our notation, but tried to keep them as similar as possible.

The join, participation and submission phases are illustrated in Fig. 3. The participation proof share that will be constructed in the participation phase is illustrated in Fig. 2.

Setup: $(spk, ssk) \leftarrow$ Setup. The setup phase is for the CA to set up all the needed keys. The CA generates a service

DBS.Prove$(g, q, \alpha, A = g^\alpha)$:		DBS.Verify(g, q, A):
	Setup	
$\rho \xleftarrow{\$} \mathbb{Z}_q, R \leftarrow g^\rho$	$\xrightarrow{R}$	
	$\xleftarrow{c_0, c_1}$	$c_0 \xleftarrow{\$} \{0,1\}^k, c_1 \xleftarrow{\$} \{0,1\}^k$
$s_0 \leftarrow \rho + c_0\alpha \pmod q$		$b \xleftarrow{\$} \{0,1\}$
$s_1 \leftarrow \rho + c_1\alpha \pmod q$		Prepare $B \in \{0,1\}^{\lceil \log_2 q \rceil + l}$, with $\lceil \log_2 q \rceil$ bits set to b.
	Distance-bounding: $\forall i : 0 \leq i < \lceil \log_2 q \rceil + l$	
	$\xleftarrow{b_i}$	$b_i \leftarrow B[i]$
$r_i \leftarrow s_{b_i}[i]$	$\xrightarrow{r_i}$	Record Δt_i
	Verification	
		Construct $r = s_b$ as the concatenation of r_i's for which $b_i = b$.
		$R \stackrel{?}{=} g^r A^{c_b}$

Fig. 1. One-round protocol instance of the DBS.Prove $\leftrightarrow$ DBS.Verify DB Schnorr protocol for $\mathrm{PK}\{(\alpha) : A = g^\alpha\}$. The protocol should be repeated in full to achieve the desired knowledge and distance-bounding errors.

public-private key-pair (spk, ssk) as follows. Let $G = \langle g \rangle$ be a group with generator g and prime order q. Let G_T be a group such that there is a bilinear pairing $e\colon G \times G \to G_T$[5]. Choose $x, y, z \xleftarrow{\$} \mathbb{Z}_q$ uniformly at random and set $X = g^x, Y = g^y, Z = g^z$. Then the service public key $spk = (q, G, G_T, g, X, Y, Z)$ and the service private key $ssk = (x, y, z)$.

Registration: $sk \leftarrow \langle \mathsf{Reg}_P(spk); \mathsf{Reg}_{CA}(ssk) \rangle$. During the registration phase, each protester will generate a secret key and obtain a signature on it by the CA — without revealing it to the CA. Thus each protester will have a signed secret key, the CA will issue only one signature per protester but will not know who has which key. The protester chooses $k, r \xleftarrow{\$} \mathbb{Z}_q$ uniformly randomly and sets $M = g^k Z^r$ and sends M to the server. The protester acts as prover and the CA as verifier while they perform the following ZKPK protocol:

$$\mathrm{PK}\{(k, r) : M = g^k Z^r\}.$$

If the proof succeeds, the CA chooses $a \xleftarrow{\$} \mathbb{Z}_q^*$ and sets $A = g^a$. Then it forms the signature $\sigma = (A, B = A^y, Z_B = B^z = Z^{ay}, C = A^x M^{axy})$ and sends it to the protester. The protester verifies that the following equations hold:

$$\begin{aligned} A &\neq 1 \\ e(g, B) &= e(Y, A) \\ e(g, Z_B) &= e(Z, B) \\ e(g, C) &= e(X, A) e(X, B)^k e(X, Z_B)^r. \end{aligned}$$

Upon success, the protester uses $sk = (\sigma, k, r)$.

The setup and registration phases can be done once and the keys reused for an arbitrary number of protests. Thus it would be suitable to have the registration phase as part of the issuance of e.g., national identity cards, passports or electronic IDs (eIDs).

[5]I.e., $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ and if $G = \langle g_1 \rangle = \langle g_2 \rangle$ then $G_T = \langle e(g_1, g_2) \rangle$.

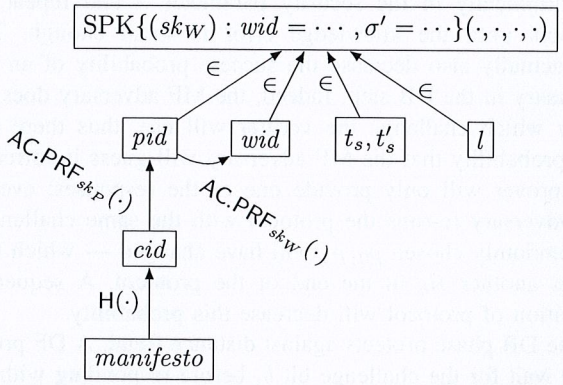

Fig. 2. Structure of a proof share. The protest (cause) identifier cid is the hash value of the manifesto. The protester P's identifier pid is computed using the protester's key sk_P and cid. The witness W's identifier wid is computed using the witness's key sk_W and pid. t_s, t'_s are the last head of the blockchain seen by the protester and witness, respectively, and l is an area. All values are signed by the witness while also proving the correctness of wid and knowledge of a signature on sk_W.

Creating a protest: the manifesto. The organizer will write a manifesto for the protest, i.e., to capture the cause. This manifesto is simply any intelligible text. He will distribute this manifesto to people: on the Web, on placards, etc. If they agree, they will use it to join the protest.

Joining participant: $(pid, t_s) \leftarrow \mathsf{Join}_P(manifesto)$. A participant who wants to join the protest will use the manifesto to compute an identifier for the cause, $cid \leftarrow \mathsf{H}(manifesto)$. Then this identifier is used to create the participant's protest-specific identifier, $pid \leftarrow \mathsf{AC.PRF}_{sk}(cid)$ (in Figs. 2 and 3). More specifically, $pid = g_T^{1/(k+cid)}$ (remember that $sk = (\sigma, k, r)$). The participant should also get a time-correlated random value from the time-stamping service, $t_s \leftarrow \mathsf{TS.Get}$.

Joining witness: $t'_s \leftarrow \mathsf{Join}_W$. The witness should simply get a time-correlated random value from the time-stamping service, $t'_s \leftarrow \mathsf{TS.Get}$. (We do this for redundancy, the newest

① Implies peaks in [redacted] usages of TS, you could be limited by the throughput of TS (bitcoin 7/s, omniledger 10^3/s). You can mention that need like.
② Having witnesses increase that need. Would be good to give an esempio of throughput need for a protest of size X (with X realistic, using real life esempio)

$O \to$ all: manifesto

P: $t_s \leftarrow$ TS.Get
$cid \leftarrow \mathsf{H}(\text{manifesto})$,
$pid \leftarrow \mathsf{AC.PRF}_{sk_P}(cid)$

W: $t'_s \leftarrow$ TS.Get

Join

$P \to W$: pid

Participation

$P \leftrightarrow W$: $\mathrm{DBPK}\{(sk_P):$
$pid = \mathsf{AC.PRF}_{sk_P}(cid)$,
$\sigma'_P = \mathsf{AC.BlindSig}(\mathsf{AC.Sign}_{ssk}(sk_P))\}$

W: $wid \leftarrow \mathsf{AC.PRF}_{sk_W}(pid)$

$W \to P$: (wid, t'_s, l)

Participation

Submission

P: $t_e \leftarrow \mathsf{TS.Stamp}(\mathsf{H}(pid, wid, t_s, t'_s, l))$

W: $t'_e \leftarrow \mathsf{TS.Stamp}(\mathsf{H}(pid, wid, t_s, t'_s, l))$

$W \to S$: $(pid, wid, t_s, t'_s, t_e, l, \pi_{wid})$, where
$\pi_{wid} = \mathrm{SPK}\{(sk_W):$
$wid = \mathsf{AC.PRF}_{sk_W}(pid)$,
$\sigma'_W = \mathsf{AC.BlindSig}(\mathsf{AC.Sign}_{ssk}(sk_W))\}$
(pid, wid, t_s, t'_s, l)

$P \to S$: $(cid, pid, wid, t_s, t'_s, t_e, l, \pi_{pid})$, where
$\pi_{pid} = \mathrm{SPK}\{(sk_P):$
$pid = \mathsf{AC.PRF}_{sk_P}(cid)$,
$\sigma'_P = \mathsf{AC.BlindSig}(\mathsf{AC.Sign}_{ssk}(sk_P))\}$
$(cid, pid, wid, t_s, t'_s, l)$

Fig. 3. An overview of of the Join, Participation and Submission phases of PRIVO. The organizer O broadcasts the manifesto. The protester P, witness W and their computations are as in Fig. 2. Finally, both P and W submits the proof share to some permanent storage S.

TODO DONE

of t_s and t'_s will set the start of the time interval of creation for the proof share.).

Participating: $\pi \leftarrow \langle \mathsf{Prticip}(cid, sk_P); \mathsf{Witness}(sk_W, spk) \rangle$, In the participating phase the participant and the witness constructs the proof share of the participant (Fig. 2). The participant reblinds its signature σ on its secret key: choose $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_q^*$ and let $\tilde{\sigma} = (\tilde{A} = A^{r_1}, \tilde{B} = B^{r_1}, \tilde{Z}_B = Z_B^{r_1}, \hat{C} = C^{r_1 r_2})$. (It is not necessary to compute a new $\tilde{\sigma}$ more than once per pid.) The participant sends $(pid, \tilde{\sigma})$ to the witness. The witness verifies that

$$\tilde{A} \neq 1$$
$$e(g, \tilde{B}) = e(Y, \tilde{A})$$
$$e(g, \tilde{Z}_B) = e(Z, \tilde{B}).$$

Both the participant and the witness compute

$$v = e(g, \hat{C})$$
$$v_x = e(X, \tilde{A})$$
$$v_{xy} = e(X, \tilde{B})$$
$$v'_{xy} = e(X, \tilde{Z}_B).$$

Then they run the following DBPK with the participant as the prover and the witness as the verifier:

$$\mathrm{DBPK}\left\{(k, r, r') : v^{r'} = v_x v_{xy}^k v_{xy}'^r \wedge pid = g_T^{1/(k+cid)}\right\}.$$

where $r' = 1/r_2$. We can rewrite $pid = g_T^{1/(k+cid)}$ as $pid^k = g_T g_T^{-cid}$. Then the PK can be designed as follows (remember that this PK should be run as a DBPK and be repeated multiple times to achieve security). The participant chooses $r_k, r_r, r_{r'} \xleftarrow{\$} \mathbb{Z}_q$ uniformly randomly, sets $R_1 = v^{r_{r'}} v_{xy}^{r_k} v_{xy}'^{r_r}$ and $R_2 = pid^{r_k}$ and sends R_1, R_2 to the witness. The witness replies with a challenge c. The participant then computes

$$s_k = -ck + r_k,$$
$$s_r = -cr + r_r,$$
$$s_{r'} = cr' + r_{r'}$$

and sends them to the witness. The witness checks whether

$$v_x^c R_1 \stackrel{?}{=} v^{s_{r'}} v_{xy}^{s_k} v_{xy}'^{s_r},$$
$$\left(g_T pid^{-cid}\right)^{-c} R_2 \stackrel{?}{=} pid^{s_k}.$$

If the proof succeeds the witness will compute $wid \leftarrow \mathsf{AC.PRF}_{sk_W}(pid) = g_T^{1/(k_W + pid)}$ (as above). Then the witness returns (wid, t'_s, l) to the participant. Now both the participant and the witness have the tuple $\pi = (pid, wid, t_s, t'_s, l)$ which constitutes a proof share of the participant (Fig. 2).

Submission: $\pi' \leftarrow \mathsf{Submit}_P(cid, pid, wid, t_s, t'_s, l)$. The participant should as soon as possible commit the proof-share data to the time-stamping service and receive the proof of commitment, $t_e \leftarrow \mathsf{TS.Stamp}(\mathsf{H}(pid, wid, t_s, t'_s, l))$. The remaining operations are not time critical. The participant computes a non-interactive zero-knowledge (NIZK) proof π_{pid} which shows the correctness of pid. More specifically,

$$\pi_{pid} \leftarrow \mathrm{SPK}\{(sk_P):$$
$$pid = \mathsf{AC.PRF}_{sk_P}(cid) \quad \wedge$$
$$\sigma'_P = \mathsf{AC.BlindSig}(\mathsf{AC.Sign}_{ssk}(sk_P))\}$$
$$(cid, pid, wid, t_s, t'_s, l).$$

Finally, the participant uploads the tuple

$$(cid, pid, wid, t_s, t'_s, t_e, l, \pi_{pid})$$

for permanent storage. Daniel ▶*Return π', as specified as output.*◀

TODO DONE

Submission: $\pi'' \leftarrow \mathsf{Submit}_W(pid, wid, t_s, t'_s, l)$. The witness should, just as the participant, commit the proof-share data to the time-stamping service, $t'_e \leftarrow \mathsf{TS.Stamp}(\mathsf{H}(pid, wid, t_s, t'_s, l))$. (This is for redundancy

① Tricky with high numbers of stamps concurrently, not all might fit in the same block ⇒ lag. The system must be flexible wit lag.

only.) Then, without any time requirements, the witness computes a NIZK proof π_{wid} as follows:

$$\pi_{wid} \leftarrow \mathrm{SPK}\,\{(sk_W) : wid = \mathsf{AC.PRF}_{sk_W}(pid) \;\wedge\; \sigma'_W = \mathsf{AC.BlindSig}(\mathsf{AC.Sign}_{ssk}(sk_W))\}(pid, wid, t_s, t'_s, l).$$

Finally, the witness uploads the tuple

$$(pid, wid, t_s, t'_s, t'_e, l, \pi_{wid})$$

for permanent storage. Daniel ▶*Return π'', as specified as output.*◀

VII. Security and Privacy Analysis

A. Individual and universal verifiability

Requirement V3 requires that Alice and Bob, as participants, can verify that their participation proofs (proof shares) are indeed included. As all proof shares (i.e., $cid, pid, wid, t_s, t'_s, t_e, t'_e, l, \pi_{pid}, \pi_{wid}$) are committed to the blockchain and available from public storage (see Fig. 3). Alice and Bob can simply check that all of their proof shares are indeed there and thus the security of individual verifiability depends on the properties of the blockchain and storage.

Requirement V2 requires that also anyone can verify the result and that all participation proofs counted indeed are eligible. As the proof shares are committed and stored publicly, anyone can download them, verify eligibility (i.e., verify π_{pid}, π_{wid}) the proofs and count them. As for individual verifiability, the security of universal verifiability is reduced to the properties of the blockchain and storage but also the eligibility verification, which we will discuss next.

B. Eligibility verifiability

Requirement V1 states that anyone must be able to determine the authenticity of the relevant attributes of the data. We have several attributes that must be verifiable: time of creation (temporal eligibility), physical location of sk_P at creation (spatial eligibility), recognize two proofs from the same person (one-proof-per-person eligibility) and that the proof is indeed designated for the event (designated-event eligibility). We will now analyse these.

a) Temporal eligibility: Requirement V1.1 ensures freshness, i.e., that Alice cannot simply resubmit an old proof as a new one. In general, freshness requires the prover to respond to an unpredictable challenge. The challenge here is to include the hash value at the head of the blockchain at the time of the proof's creation (included as t_s and t'_s in the proof share). As per construction of blockchains, the hash value at the head depends on the previous blocks in the chain. Thus the predictability of t_s depends on the predictability of the blocks, which depends on the predictability of the transactions and the nonce needed by the miner. The transactions in turn depend on the signatures made by the signing keys corresponding to the used transactions. If these were predictable, our adversary could spend arbitrary transactions in the blockchain — which is designed to be hard. Thus Alice can predict t_s with negligible probability, i.e., t_s must have been published on the blockchain before being included in her proof share.

According to requirement V1.1, we must also prove that a proof share has not been created after a certain time. Otherwise Grace can argue that the proof share was created after the protest. The hash values of the proof shares are committed to the blockchain, thus there is a negligible probability that they were created after that. Alice would have to choose a value y in the domain of the hash function H and then find a preimage x such that $y = \mathsf{H}(x)$ and x is a valid proof. If H is collision resistant, then finding *any* preimage is hard.

b) Linkability: Requirement V1.3 is required to prevent Sybil attacks, i.e., that Alice can provide two participation proofs and thus be counted twice. This is prevented by the use of pid. To be counted twice, Alice must produce a $pid' \neq pid$. Due to the deterministic property of AC.PRF, Alice must produce a new key sk'_P and make the witness accept in the protocol

$$\mathrm{PK}\{(sk'_P) : pid' = \mathsf{AC.PRF}_{sk'_P}(cid) \;\wedge\; \sigma''_P = \mathsf{AC.BlindSig}(\mathsf{AC.Sign}_{ssk}(sk'_P))\}$$

while she does not know a valid signature on sk'_P. This is thus reduced to the security of the AC scheme and how often Alice can get a valid signature on a secret key from the CA.

c) Spatial eligibility: Requirement V1.2 is achieved by having a witness vouch that Alice was indeed on the location when the proof share was created. Thus we must ensure that Alice cannot forge undeserved witness signatures, which is ensured using the same linkability mechanism as above. To forge a witness signature, Alice must produce a wid' such that

$$\pi_{wid'} \leftarrow \mathrm{SPK}\,\{(sk'_W) : wid' = \mathsf{AC.PRF}_{sk'_W}(pid) \;\wedge\; \sigma''_W = \mathsf{AC.BlindSig}(\mathsf{AC.Sign}_{ssk}(sk'_W))\}(pid, wid', t_s, t'_s, l),$$

while not knowing a signature on the secret key sk'_W. Thus she must break the AC scheme to succeed or get a valid signature on sk'_W from the CA. (Note that Alice can produce one witness signature for herself, since she has access to her own sk_P, but no more.)

d) Designated protest: Requirement V1.4 is to prevent Alice from reusing the same proof (or proof share) for another event — or someone else using her proof-share for another event. This is prevented with the use of cid in the proof shares. Thus to reuse the proof share for another protest, with a different manifesto, one must find a second preimage $manifesto'$ such that $cid = \mathsf{H}(manifesto) = \mathsf{H}(manifesto')$.

C. Privacy

We start with requirement P2, that Alice's proofs must be unlinkable across protests (but linkable within one protest due to requirement V1.3). It follows from the properties of AC.PRF that $pid = \mathsf{AC.PRF}_{sk_P}(cid)$ and $pid' = \mathsf{AC.PRF}_{sk_P}(cid')$ are unlinkable if $cid \neq cid'$.

①② It's a lot of text about the downsides/unfeasability of this solution.
③⑤ Can be merged, feels very similar

Now to requirement P1. Given pid, Grace cannot distinguish whether $pid = \mathsf{AC.PRF}_{sk_A}(cid)$, $pid = \mathsf{AC.PRF}_{sk_A}(cid')$ or $pid = \mathsf{AC.PRF}_{sk_B}(cid)$ due to the properties of AC.

However, say that a witness works for Grace. This witness interacts with Alice directly, i.e., has (physically) identified her. The witness also learns Alice's pid, but not cid, to compute wid and to verify the DBPK. Daniel ▶*This assumes that I will do the changes to hide cid from the witness in the DBPK.*◀ Afterwards, Alice publishes pid and cid together as part of the proof share. The probability of a colliding $pid = \mathsf{AC.PRF}_{sk}(cid')$ for some $cid' \neq cid$ and $sk \neq sk_P$ is "small". Daniel ▶*Fix this.*◀ Now Grace can link cid (and thus the opinions in the manifesto) to Alice.

VIII. Performance and Implementation

A. Implementation on smartphones or smart-cards

Technological progress has enabled more advanced cryptography on Smart-cards and Smartphones which could be used to implement our solution : Benchmarks [43] have shown that Android devices are now fast enough to efficiently implement Privacy Enhancing Technologies (PETs), with a Samsung Galaxy S i9000 being able to execute Idemix in 153 ms. However, those benchmarks also demonstrate that SmartCards remain slow to process complex protocols such as Idemix or U-Prove (taking between 4 s and 8 s to process them). While the limited processing power of many embarked systems has been a challenge, Idemix has been succesfully implemented to prove the posession of credentials on Java Cards by Bichsel et al. in 2009 [44] and the IRMA project, released in 2014, aimed to achieve an implementation « suitable for real life transactions » [45] while maintaining security and privacy for its users.

B. Threshold of witnesses

Assume that every person has 5000 contacts in their contact book [46]. Then it would be reasonable to set the threshold of at least 6000 witnesses. According to the Anon-Pass performance measures [33] each witness signature would require 8 ms per core on a quad-core Intel 2.66 GHz Core 2 processor. This yields $8\,\text{ms} \times 6000 = 48\,000\,\text{ms} = 48\,\text{s}$. of processing.

~~DONE TODO~~ IX. Conclusions

We have introduced PRIVO, a protocol for verifiably counting participants at protests. Despite the verification, the privacy of the participants is preserved to the extent possible by their physical presence at the protest. To achieve that, we adapted the Schnorr protocol to develop distance-bounding anonymous credentials. Except for receipt freeness, which we cannot guarantee if the adversary gets access to Alice's device and secret PRF key after her participation has been witnessed and before she had a chance to delete it, PRIVO fulfills all verification and privacy properties we derived from those of electronic voting complemented by the specific constraints on time and location required for protests. In general, none of the related works fulfills the eligibility verifiability requirements V1.1, V1.2 and V1.3. To some extent, some techniques do fulfill requirement V1.4, since people usually use placards in the protests. Also, some schemes provide universal verifiability (requirement V2), but not individual verifiability (requirement V3). If they would provide individual verifiability, they would likely violate the privacy requirements.

Finally, most of the current methods are estimates, with quite some error margins. While PRIVO is an actual count and not an estimate, its accuracy for the total number of participants hinges on the participants having the necessary equipment (a smartphone or similar device), some sort of trustworthy credential, and the willingness to run the protocol. If used today, PRIVO would result in a considerable undercount in most scenarios, Daniel ▶*why?*◀ but it presents a first step toward accurate verifiable yet privacy-preserving crowd counting by showing how it can be done in theory at least and in practice once some assumptions become more realistic.

When it comes to receipt freeness, our construction does not allow it. The very property that prevents Alice from cheating by creating pid, pid', such that $pid \neq pid'$, is the property that allows Grace to verify Alice's participation proof. If Grace can perform computations using Alice's key, Grace can input the cid of interest and see if the resulting pid is available on the blockchain. To mitigate this situation, Alice could delete her key, in which case she cannot participate in any future protest.

As implied by the last statement, there is the problem of renewing credentials. As with all identity cards, passports etc., they have a limited lifetime and must be renewed at regular intervals. The rate at which Alice is allowed to renew her credentials affects to what extent she can perform Sybil attacks.

Currently, the only problem preventing wide deployment is the lacking hardware in smartphones. However, due to the rapid development of ~~"smart"~~ electronic solutions, we believe this is only a matter of time. E.g., mobile payments and public transport tickets that can be used over Near-Field Communication will eventually require a DB chip. Without DB for these applications, the user must manually confirm on the screen to prevent attacks. We believe that the drive for increased usability will replace this type of confirmation by distance bounding.

The deployment of the needed credentials should not be a problem either. There are already countries where national eID systems are widely deployed. The speed of adoption will depend on the architecture of these systems. For instance, the system used in Sweden, BankID, is implemented in software and can thus be easily upgraded to include the anonymous credentials we need. As for coverage of the population: more than 95 % of people in the ages 21–50 uses BankID, it is 88 % for ages 51–60 and 76 % for ages 61–70[6].

While the deployment of the necessary technologies are not yet available in authoritarian regimes, they are in several democracies. We believe that it as important to implement systems such as ours to prevent any democracy of slipping down the slope towards dictatorship.

[6]Official statistics, in Swedish: https://www.bankid.com/assets/bankid/stats/2018/statistik-2018-04.pdf.

① She could impact the timestamping service?

Finally, Mallory represents another nation state and has some interest in affecting the stability of Grace's regime, for Mallory's own gain, thus supporting either Grace or Alice as she see fits. Thus, the objective of Mallory will also be to either increase or decrease the count. Seb ►*we have to justify why Mallory is not captured either by Alice or Grace*◄ In addition, Mallory could also have simply as her objective to cause a denial-of-service attack on the architecture of PRIVO.

Mallory cannot create keys valid in another nation state.

References

[1] K. Tong-Hyung and Y. Lee, "Counting 1 million crowds at anti-president rallies in Seoul," *Associated Press: The Big Story*, Nov. 2016. [Online]. Available: http://bigstory.ap.org/article/317ea62bddbd4132ab1467863a532ab9/counting-1-million-crowds-anti-president-rallies-seoul (visited on 04/05/2017).

[2] R. Meyer, "How will we know trump's inaugural crowd size?" *The Atlantic*, Jan. 20, 2017, ISSN: 1072-7825. [Online]. Available: https://www.theatlantic.com/technology/archive/2017/01/how-will-we-know-trumps-inaugural-crowd-size/513938/ (visited on 07/24/2017).

[3] K. Waddell, "The exhausting work of tallying America's largest protest," *The Atlantic*, Jan. 2017, ISSN: 1072-7825. [Online]. Available: https://www.theatlantic.com/technology/archive/2017/01/womens-march-protest-count/514166/ (visited on 04/05/2017).

[4] Al-Jazeera. (May 21, 2017). Huge marches as venezuela marks 50 days of protest, [Online]. Available: http://www.aljazeera.com/news/2017/05/venezuelan-opposition-marks-50-days-protests-170520174956348.html (visited on 08/01/2017).

[5] S. Edwards, *Barcelona protesters demand release of jailed separatist leaders*, [Online; accessed 2. Feb. 2018], Nov. 11, 2017. [Online]. Available: https://www.independent.co.uk/news/world/europe/barcelona-protesters-demand-release-of-jailed-separatist-leaders-catalonia-latest-a8050116.html (visited on 02/02/2018).

[6] C. P. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptology*, vol. 4, no. 3, 1991. DOI: 10.1007/BF00196725.

[7] S. Brands and D. Chaum, "Distance-bounding protocols," in *Workshop on the Theory and Application of of Cryptographic Techniques*, Springer, 1993, pp. 344–359.

[8] C. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun, "Distance hijacking attacks on distance bounding protocols," in *Security and Privacy (SP), 2012 IEEE Symposium on*, IEEE, 2012, pp. 113–127.

[9] Y. Desmedt, "Major security problems with the 'unforgeable'(feige)-fiat-shamir proofs of identity and how to overcome them," in *Proceedings of SECURICOM*, vol. 88, 1988, pp. 15–17.

[10] S. Brodzinsky, "Venezuela to vote on constituent assembly after months of protests," *The Observer*, Jul. 29, 2017, ISSN: 0029-7712. [Online]. Available: http://www.theguardian.com/world/2017/jul/29/venezuela-government-maduro-vote-end-democracy (visited on 08/01/2017).

[11] S. Gambs, C. Eduardo Rosar Kos Lassance, and C. Onete, "The Not-so-distant Future: Distance-Bounding Protocols on Smartphones," in *14th Smart Card Research and Advanced Application Conference*, Bochum, Germany, Nov. 2015.

[12] A. Taylor. (Jun. 12, 2017). Months of deadly anti-government protests in venezuela - the atlantic, [Online]. Available: https://www.theatlantic.com/photo/2017/06/months-of-anti-government-protests-continue-in-venezuela/530031/ (visited on 08/01/2017).

[13] J. R. Douceur, "The sybil attack," in *Peer-to-Peer Systems*, 2002. DOI: 10.1007/3-540-45748-8_24.

[14] Briar. (2016). Briar: A messaging app designed for activists, [Online]. Available: https://briarproject.org/how-it-works.html (visited on 02/25/2018).

[15] FireChat. (2016). Firechat: A messaging app without internet access or cellular data, [Online]. Available: https://www.opengarden.com/firechat.html (visited on 02/25/2018).

[16] U. N. Kar and D. K. Sanyal, "An overview of device-to-device communication in cellular networks," *ICT Express*, 2017, ISSN: 2405-9595. DOI: 10.1016/j.icte.2017.08.002.

[17] S. Delaune, S. Kremer, and M. Ryan, "Verifying privacy-type properties of electronic voting protocols," *Journal of Computer Security*, vol. 17, no. 4, pp. 435–487, 2009.

[18] BBC Magazine, "Protest numbers: How are they counted?" *BBC News*, Mar. 28, 2011. [Online]. Available: http://www.bbc.com/news/magazine-12879582 (visited on 07/24/2017).

[19] F. El-Baz. (Jun. 2003). The [?]-man march, WIRED, [Online]. Available: https://www.wired.com/2003/06/crowd-spc/ (visited on 07/21/2017).

[20] M. Goldberg. (Jan. 2003). The protest-crowd numbers game - salon.com, Salon, [Online]. Available: http://www.salon.com/2003/01/24/crowds/ (visited on 07/21/2017).

[21] CrowdSize. (2016). Crowdsize iPhone application, [Online]. Available: http://www.crowdsize.com/ (visited on 07/24/2017).

[22] C. Zhang, H. Li, X. Wang, and X. Yang, "Cross-scene crowd counting via deep convolutional neural networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 833–841. DOI: 10.1109/CVPR.2015.7298684.

[23] M. Vanhoef, C. Matte, M. Cunche, L. S. Cardoso, and F. Piessens, "Why MAC address randomization is not enough: An analysis of wi-fi network discovery mechanisms," in *Proceedings of the 11th ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '16, New York, NY, USA: ACM, 2016,

pp. 413–424, ISBN: 978-1-4503-4233-9. DOI: 10.1145/2897845.2897883. [Online]. Available: http://doi.acm.org/10.1145/2897845.2897883 (visited on 07/24/2017).

[24] R. Melina. (Sep. 2010). How is crowd size estimated? Live Science, [Online]. Available: https://www.livescience.com/8578-crowd-size-estimated.html (visited on 07/20/2017).

[25] CrowdCount. (2016). Crowd count: Real-time crowd sizing, [Online]. Available: http://crowdcount.org/ (visited on 04/05/2017).

[26] P. Danielis, S. T. Kouyoumdjieva, and G. Karlsson, "Urbancount: Mobile crowd counting in urban environments," in *8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), October 03-05, 2017, Univ British Columbia, Vancouver, Canada*, Institute of Electrical and Electronics Engineers (IEEE), 2017, pp. 640–648.

[27] ——, "Divote: A distributed voting protocol for mobile device-to-device communication," in *Teletraffic Congress (ITC 28), 2016 28th International*, IEEE, vol. 1, 2016, pp. 69–77.

[28] S. Gambs, M.-O. Killijian, M. Roy, and M. Traore, "Props: A privacy-preserving location proof system," in *Reliable Distributed Systems (SRDS), 2014 IEEE 33rd International Symposium on*, 2014. DOI: 10.1109/SRDS.2014.37.

[29] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups," in *Annual International Cryptology Conference, CRYPTO'97*, Springer, 1997, pp. 410–424.

[30] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology — CRYPTO' 86: Proceedings*. 1987. DOI: 10.1007/3-540-47721-7_12.

[31] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Annual International Cryptology Conference*, 1991, pp. 129–140.

[32] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Annual International Cryptology Conference*, Springer, 2004, pp. 56–72.

[33] M. Z. Lee, A. M. Dunn, B. Waters, E. Witchel, and J. Katz, "Anon-pass: Practical anonymous subscriptions," in *Security and Privacy (SP), 2013 IEEE Symposium on*, IEEE, 2013, pp. 319–333.

[34] Y. Dodis and A. Yampolskiy, "A verifiable random function with short proofs and keys," in *International Workshop on Public Key Cryptography*, Springer, 2005, pp. 416–431.

[35] J. Camenisch, S. Hohenberger, M. Kohlweiss, A. Lysyanskaya, and M. Meyerovich, "How to win the clonewars: Efficient periodic n-times anonymous authentication," in *13th ACM Conference on Computer and Communications Security*, 2006, pp. 201–210. DOI: 10.1145/1180405.1180431.

[36] Y. Desmedt, C. Goutier, and S. Bengio, "Special uses and abuses of the fiat-shamir passport protocol," in *Conference on the Theory and Application of Cryptographic Techniques*, Springer, 1987, pp. 21–39.

[37] G. Avoine and A. Tchamkerten, "An efficient distance bounding rfid authentication protocol: Balancing false-acceptance rate and memory requirement," in *International Conference on Information Security*, Springer, 2009, pp. 250–261.

[38] I. Boureanu, A. Mitrokotsa, and S. Vaudenay, "Practical and provably secure distance-bounding," *Journal of Computer Security*, vol. 23, no. 2, pp. 229–257, 2015.

[39] U. Dürholz, M. Fischlin, M. Kasper, and C. Onete, "A formal approach to distance-bounding rfid protocols," in *International Conference on Information Security*, Springer, 2011, pp. 47–62.

[40] S. Vaudenay, "Sound proof of proximity of knowledge," in *International Conference on Provable Security*, Springer, 2015, pp. 105–126.

[41] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, *Omniledger: A secure, scale-out, decentralized ledger via sharding*, Cryptology ePrint Archive, Report 2017/406, To appear in 39th IEEE S&P 2018, 2017.

[42] S. Nakamoto, *Bitcoin: A peer-to-peer electronic cash system*, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

[43] J. Hajny, L. Malina, Z. Martinasek, and O. Tethal, "Performance evaluation of primitives for privacy-enhancing cryptography on current smart-cards and smart-phones," in *Data Privacy Management and Autonomous Spontaneous Security*, Springer, 2014, pp. 17–33.

[44] P. Bichsel, C. Binding, J. Camenisch, T. Groß, T. Heydt-Benjamin, D. Sommer, and G. Zaverucha, "Cryptographic protocols of the identity mixer library," *Tech. Rep. RZ 3730, Tech. Rep.*, 2009.

[45] A. De La Piedra, J.-H. Hoepman, and P. Vullers, "Towards a full-featured implementation of attribute based credentials on smart cards," in *International Conference on Cryptology and Network Security*, pp. 270–289.

[46] M. Marlinspike. (Jan. 2014). The difficulty of private contact discovery, [Online]. Available: https://whispersystems.org/blog/contact-discovery/.

FIX appeared

[47] "Short paper: service-oriented sharding for blockchains", Gencer, van Renesse, Gün Sirer