2013 IEEE 32nd International Symposium on Reliable Distributed Systems

A Distributed Polling with Probabilistic Privacy

Yahya Benkaouz*, Rachid Guerraoui§, Mohammed Erradi*, Florian Huc§
* *Networking and Distributed Systems Research Group, TIES, SIME Lab, ENSIAS*
Mohammed V-Souissi University, Rabat, Morocco
y.benkaouz@um5s.net.ma, erradi@ensias.ma
§*Distributed Programming Laboratory*
EPFL, Lausanne, Switzerland
rachid.guerraoui@epfl.ch, florian.huc@epfl.ch

***Abstract*—In this paper, we present PDP, a distributed polling protocol that enables a set of participants to gather their opinion on a common interest without revealing their point of view. PDP does not rely on any centralized authority or on heavyweight cryptography. PDP is an overlay-based protocol where a subset of participants may use a simple sharing scheme to express their votes. In a system of M participants arranged in groups of size N where at least $2k-1$ participants are honest, PDP bounds the probability for a given participant to have its vote recovered with certainty by a coalition of B dishonest participants by $\pi(B/N)^{k+1}$, where π is the proportion of participants splitting their votes, and k a privacy parameter. PDP bounds the impact of dishonest participants on the global outcome by $2(k\alpha + BN)$, where α represents the number of dishonest nodes using the sharing scheme.**

***Keywords*-Privacy; Distributed Polling; PDP.**

I. Introduction

The privacy has become one of the top concerns and requirements of many distributed collaborative applications. Collaborative tasks and computations are often conducted based on the inputs supplied by the collaborative users. Such computations could occur between trusted partners, between partially trusted partners, or between competitors [2]. Online Social Networks 'OSN' with up to a billion users have dramatically raised concerns on privacy leakage. As far as privacy is concerned, users' personal data should be accessible only to authorized users within such networks. Recall that an OSN could be defined as a web-based platform that allows users to share a variety of personal-related information including: Ideas, activities, events and interests within their individual networks. The user's individual networks are made up of individuals tied by one or more specific types of interdependency, such as friendship, common interest or other relationships. Hence, an individual network consists in nodes (i.e. users) connected by different types of ties which represent the relationships between such nodes.

In some kind of social network application, even directly connected nodes should not have access to various bits of private information. Consider the following example of polling within OSN: The users of a social network use a polling to take decisions about political, professional or social interests (e.g. the organizers of a Saturday night party asking in a social group whether partners should be invited too?). While participants care more and more about their reputation, no one likes to disclose his point of view to the other participants. Therefore, such groups need a way to launch polling while maintaining the privacy of participant's point of view.

As authors of [1] point out, an easy way to conduct a poll is to use a central server. Each participant sends its vote to a central entity, which subsequently aggregates all votes and computes the outcome. Besides the non-scalability of this solution, the privacy is not ensured as participants might generally not want their votes (and maybe even the subject of the poll or the result) to be seen by a central entity, be it trusted or not.

The above example could be seen as two or more parties wanting to conduct a computation based on their private inputs, but neither party is willing to disclose its own input to anybody else. The problem is how to conduct such a computation while preserving the privacy of the inputs. This problem is referred to as Secure Multi-party Computation problem 'SMC' in the literature [3].

Currently, to solve the above problems, a common strategy is to assume the trustworthiness of the service providers, or to assume the existence of a trusted third party, which is risky in nowadays' dynamic and malicious environment [2]. Therefore, protocols that can support joint computations while protecting the participants' privacy are of growing importance. In theory, the general secure multi-party computation problem is solvable [3]. However, using the solutions derived by these general results for special cases of multi-party computation can be impractical; special solutions should be developed for special cases for efficiency reasons [4].

In this paper, we consider a set of M participants $P = \{p_1, p_2, ..., p_m\}$, involved in a polling session (e.g. a set of persons wish to launch a survey on a common interest). Each participant will vote to express his own opinion. In general, we denote v_i the vote of the participant p_i. After the execution of the polling protocol, the output consists in the sum of participants' votes. The computed sum represents

1060-9857/13 $26.00 © 2013 IEEE
DOI 10.1109/SRDS.2013.13

the tendency of the majority of participants. We consider a fully decentralized scheme where there is neither central trusted server nor an entity with a specific role. Hence, all participants are involved in the computation of the polling outcome. We consider polling within two choices. Participants choose as input one of two values (e.g. (yes or no), (agree or not agree), (support or against) etc).

Participants could either be honest or dishonest. An honest participant strictly follows the protocol and contributes to the verifications as long as its privacy is not compromised. A dishonest participant may misbehave to reveal the opinion of honest participants. An important issue is when a group of dishonest participants, named coalition, conspires in order to reveal the input of a given participant. We aim at enabling participants to have precise polling's output. At the same time, no user should learn anything about the input of other honest users, even if he colludes with the other malicious users.

The key contribution of this paper consists in PDP, a probabilistic Private Distributed Polling protocol that allows a set of users to conduct a polling session without revealing their choices. PDP ensures votes privacy such that, in a system of M participants organized in groups of size N where at least $2k-1$ participants are honest, PDP bounds the probability for a given participant to have its vote recovered with certainty by a coalition of B dishonest participants by $\pi(B/N)^{k+1}$, where π is the proportion of participants splitting their votes, and k a privacy parameter. Considering the uncertain detection where a coalition of dishonest participants agrees on a probabilistic detection rule (section VI. B), the probability of votes detection is represented by the following formula : $\frac{B}{N}[\frac{N-r}{N} + r(k+1)\prod_{i=0}^{B-1}\frac{N-2k-i}{N-i}]$. Further, PDP bounds the impact of dishonest participants on the global outcome by $2(k\alpha + BN)$, where α is the number of dishonest nodes using the sharing scheme. To sum up, our protocol has the following advantages: (1) The protocol's output is computed by participants themselves without interaction with any third entity; (2) The protocol preserves data privacy such that the probability of vote detection is related to the decision rule used by dishonest participants; (3) The polling's outcome is accurate when there is no data loss or dishonest act; (4) The impact of dishonest participants on the outcome is bounded. Moreover, PDP does neither rely on heavyweight cryptographic techniques nor on the trustworthiness of a third party. Reducing the usage of cryptographic techniques helps voters to use the protocol even with low-performance equipment. It also avoids the need to manage key distribution.

Section 2 gives an overview of existing works. Section 3 describes the system model. The PDP protocol is described in Section 4. In Section 5 we study the PDP performance. In Section 6 a privacy analysis has been made. A discussion about the impact of dishonest nodes is given in Section 7 where we compute the maximum gap on poll's result due to dishonest acts. Section 8 concludes the paper.

II. Related Work

Several works have been proposed to ensure the privacy and the anonymity of voting participants. Existing works can be divided into two main categories: Protocols based on cryptographic primitives and those based on secret sharing techniques. In general, cryptographic based voting protocols make use of one the following techniques:

- Mix-net (introduced by Chaum in [5]): In [9], the authors proposed a secure anonymous channel that avoids the problem of ciphertext length expansion. Thus, they presented an election scheme based on the proposed anonymous channel. Authors of [10] proposed the Randomized Partial Checking in order to improve the robustness of mix-net based e-voting protocols.
- Blind signature (introduced by Chaum in [6]): Authors of [11] presented an electronic polling based on blind signature and RSA algorithm. The considered architecture assumes, in addition to the voters, a polling server and a trusted third party.
- Homomorphic schemes based on ElGamal [7] or Paillier [8] cryptosystems: In [12], the authors presented a multi-authority secret ballot scheme based on the ElGamal encryption. Authors of [13] proposed a voting protocol based on [12] to improve the total complexity. The ensured privacy relies on the trustworthiness of the authorities. Several protocols based on homomorphic schemes have been proposed [14-17].

The above solutions have an important computational cost. Actually, the cryptographic based protocols help to ensure data privacy. However, they are too complex and time consuming for being used in a large scale polling system. On the other hand, since the introduction of secret sharing scheme based on polynomial interpolation by Shamir in [18], several extensions and protocols have been proposed. Authors of [19] and [20] proposed respectively the Verifiable Secret Sharing scheme and a scheme to combine multiple secrets by direct operations on the shares. The later was extended in [21] where the authors showed how multiparty computation may be conducted based on VSS. In fact, these techniques may be used for polling application, but they involve higher mathematics.

Moreover, based on a variation of Chaum's mix nets, Authors of [22] proposed AMPC (Anonymous MuliParty Computation) which provides a generic building block for electronic anonymity in various applications. AMPC and enhanced check vector have been applied in [23] for e-voting application. While powerful, this protocol differs from our work since participants have distinct and predefined roles. This may result in decreased scalability and robustness if specific nodes fail. Therefore, the majority of the existing works related to voting protocols is based on cryptographic

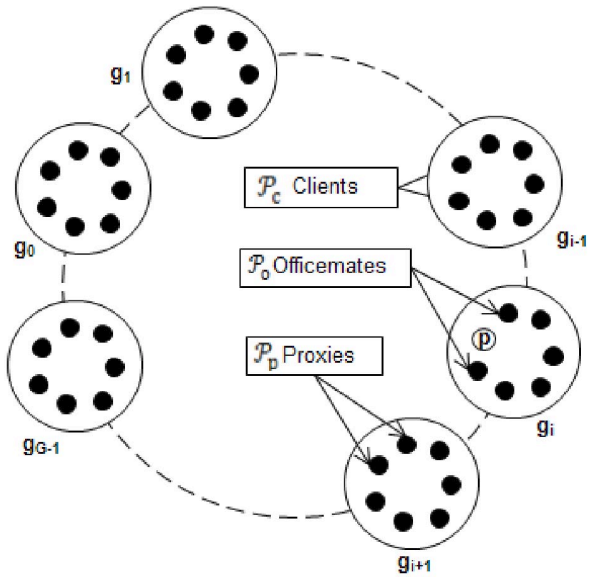

Figure 1. The system model

primitives, trusted third parties, or assign specific role to some entities.

Recently, the authors of [1] proposed DPol, a protocol for distributed polling. This protocol is based on a simple sharing scheme. In DPol, the authors do not take into account the number of participants splitting their inputs, and assume that all participants use an identical sharing scheme. Thus, more messages are exchanged between participants. In this paper, we propose a distributed polling protocol that takes into account this parameter (i.e. The number of participants splitting their inputs). We show that this reduces the number of exchanged messages and the execution time. We consider two types of detections in the privacy analysis (Section VI): Reliable and Uncertain detection of a participant's vote. PDP reduces the probability of reliable detection, whereas in the uncertain detection, malicious participants do not have any insurance that the disclosed value represents the real participant's input.

III. System Model

The model consists of M participants; each participant is represented as a uniquely identified node. Each participant has its private numerical input v_i. The global outcome of the polling protocol is the sum of votes made by the existing participants ($\sum_i v_i$).

The M nodes are clustered into G ordered groups, from g_0 from to g_{G-1}. Each group contains N_j nodes ($\sum_{j=1}^{G} N_j = M$). A node p_i in group g_j maintains two sets of nodes: A set P_o of officemates containing all nodes belonging to the same group ($P_o = g_j \backslash \{p\}$) (i.e. the set of participants in g_j except p) and a fixed-size set P_p of proxies, containing a subset of next group ($P_p \subseteq g_{i+1 mod G}$). The number of proxies for a given node is equal to 1 or $2k+1$. The procedure to define the number of proxies is described later in this section. Therefore, all groups virtually form a ring, g_0 being the successor of g_{G-1} (Fig. 1). Each group g_j is a clique. We define a client of p as a node for which p acts as a proxy. In other words, if p is a proxy of q, then q is one of its clients. Every node maintains a list of its clients in the previous group ($P_c \subseteq g_{i-1 mod G}$). A node discards every message originating from a node that is not in $P_c \cup P_o$. We assume: a random uniform distribution of nodes across the G groups, nodes in the successor groups are distributed uniformly at random as proxies in the predecessor groups and messages arrive correctly to their destinations.

An overlay, respecting such system model, may be constructed based on Distributed Hash Tables (DHT), where the group identifier of a given node is determined based on the node's id (e.g. $id_{grp} = Hash(id_{node}) mod(G)$). The number of proxies assigned to a given node is based on a computation that involves node and group identifiers (e.g. we may compute the hash function of the node identifier and the group number modulo a random value. If the result is an odd number, then one proxy will be assigned to that node, else we will associate $2k+1$ proxies). Then, each node establishes a three-way-handshake protocol to inform its proxies that it is one of their clients. Therefore, each node gets the list of its clients and proxies. The list of officemates is received by a simple request for nodes having the same group identifier. Let us note that byzantine agreement protocol might be used as subroutine to handle the problem of dishonest nodes that may fake their group membership. This issue is not treated in this work.

IV. Protocol Desription

In the following, we give a description of the suggested probabilistic Private Distributed Polling protocol (PDP). In PDP, participants are clustered in fully connected groups (or clusters). Participants may use a message sharing technique to encode their private inputs, in such a way that it is not possible to differentiate a partial message from the entire message. Then, they send the shares of their inputs to proxies, belonging to another group. Each group computes a partial tally that is further broadcasted to all other groups. Each participant eventually outputs the same global outcome.

PDP consists in three steps: Sharing, counting and broadcasting. During the first step, a participant generates a single message ($m_i = v_i$) or a set of shares reflecting the private input ($\sum_i m_i = v_i$) and sends each generated share to one of its proxies (Line 1 in Algorithm1). In the counting step, each node will compute the sum of the clients shares (Line 2 in Algorithm1). Such sum is called individual tally. Then each proxy will broadcast the result to its officemates. Each participant will compute the local tally which is the sum of the individual tallies received from its officemates (Line

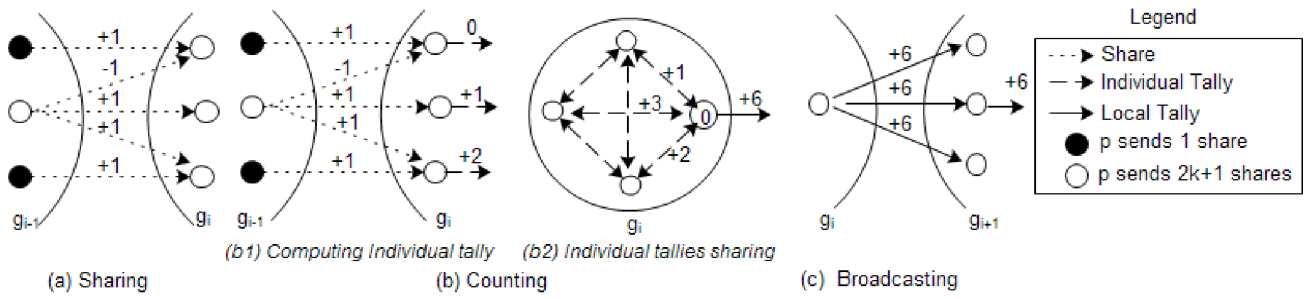

Figure 2. PDP protocol

Algorithm 1 Node p in group $g_i, i \in \{0, .., G-1\}$

Input: a vote $v \in \{-1, +1\}$
Variables:
an individual tally, $it = 0$
a local tally, $lt = 0$
an array of local tally sets, $S[\{1, .., G\}]$
a local tally array, $T[1, .., G]$
Output: the global tally gt
Polling Algorithm
1. $vote(v, P_p)$
2. $local\_count(it, P_p)$
3. $lt = lt + it$
4. $local\_tally\_broadcast(i, lt, P_p)$
5. $gt = \sum_i T[i]$

Algorithm 2 Sharing phase

Procedure $vote(v, P_p)$ **is**
1. $m = v$
2. **if** $(|P_p| == 1)$ **then**
3. $send[share, m](P_p)$
4. **else**
5. **for each** $proxy \in P_p$ **do**
6. $send([share, m], proxy)$
7. $m = -m$
8. **end for**
9. **end if**

upon event $< receive|[share, m] >$ **do**
10. $it = it + m$

3 in Algorithm1). Finally, the local tallies are forwarded along the ring so that all nodes eventually compute the final outcome, named global tally (Line 4, 5 in Algorithm 1).

Step 1: Sharing. (Fig. 2 (a)) Each participant node could cast its vote as a single share or as a set of $2k + 1$ shares. Votes are encoded in boolean value ($+1$ or -1). To split a vote into $2k + 1$ messages, the messages and their sum should belong to the set $\{-1, +1\}$ (i.e.$\forall i, m_i \in \{-1, +1\}$ and $\sum_i m_i \in \{-1, +1\}$), and $k + 1$ messages should be identical. Since the votes and their shares belong to the set $\{-1, +1\}$, participants cannot differentiate between a vote and its shares. Hence, for a participant who wants to vote for +1-value, the participant has to send $k + 1$ messages containing +1-value and the other k messages with -1-value. Let us recall that some participants could send their votes in single shares (i.e. a single message containing -1 or $+1$). Hence, when a proxy receives a message from a given client node, the proxy could not distinguish if such share was generated as a single one or it is one among the previously generated $2k+1$ shares. Once a node has generated its $2k+1$ messages, it sends each of them to a distinct proxy. Once every node in the system has received one share from each of its clients, the sharing phase is over (Algorithm 2).

Step 2: Counting. (Fig. 2 (b)) Recall that in the sharing step, each participant node within a group g_{i-1}, sends a single share or $(2k + 1)$ shares to its proxies in the group g_i. Now, in the intermediate counting step, each proxy within the receiving group g_i will compute the sum of the received shares from its clients in the group g_{i-1} (Line 10 in Algorithm 2). The value of the counted sum is designated as the Individual Tally ($IT_j = \sum_i m_i^j$, such that m_i^j represents a message sent from participantp_i to p_j). Note that each proxy will have its own individual tally. Each node waits for the reception of the expected number of shares from its clients (i.e. based on the list of his clients).

Algorithm 3 Counting phase

Procedure $local\_count(it, P_o)$ is
1. **for each** $officemate \in P_o$ **do**
2. $send([IndividualTally, it], officemate)$
3. **end for**

upon event $< receive|[IndividualTally, t] >$ **do**
4. $lt = lt + t$

Once a participant node has received the expected number of shares, it broadcasts the computed individual tally to its officemates (Lines 1-3 in Algorithm 3). Each officemate will compute the sum of the received individual tallies resulting in a local tally of its group ($LT_j = \sum_{p \in g} IT_p$) (Line 4 in Algorithm 3). Then, each officemate will forward the

computed local tally to its proxies in the next group.

Step 3: Broadcasting. (Fig. 2 (c)) In the previous step, each officemate has calculated the local tally of its group. During the local tally forwarding step, each officemate will send the local tally of its group to its proxies (Lines 1-3 in Algorithm 4). If all the officemates are honest, their computation leads to the same value of the local tally. Once a participant node in the group g_i receives the local tally of its group g_i from the client in the previous group g_{i-1}, the local tally is no longer forwarded. When a participant node receives local tallies of all groups, the global tally will be computed by summing these local tallies ($GT = \sum_{g=0}^{G-1} LT_g$). In this case, such value has crossed the ring and received in its initial sender. This global tally is the global outcome of the polling session. Note that all participants should compute the global tally after reception of local tallies.

Algorithm 4 Broadcasting phase

Procedure $local\_tally\_broadcast(i, lt, P_p)$ **is**
1. **for each** $proxy \in P_p$ **do**
2. $send([LocalTally, i, lt], proxy)$
3. **end for**

upon event $< receive|[LocalTally, i_{group}, t] >$ **do**
4. $S[i_{group}] = S[i_{group}] \cup t$
5. **if** $(S[i_{group}] = |P_c|)$ **then**
6. $local\_tally\_broadcast(i, T[i_{group}])$
7. **end if**

V. Correctness And Complexity Analysis

A. PDP's Accuracy

In order to prove the correctness of the suggested PDP algorithm, we assume that all participants are honest respecting the protocol rules. An honest participant votes using a single or $2k+1$ messages and remains conform to the rules of the PDP protocol.

Theorem: Assume a polling system with only honest participants, where each participant votes with a single message $m_i = v_i$ or $2k+1$ messages $\sum_{j=1}^{2k+1} m_j = v_i$, the PDP algorithm terminates and each participant node eventually outputs the total sum of the participants votes ($\sum_p v_p$).

Proof: (Correctness). In order to prove the accuracy of this protocol, we will start from the first phase. For a given group g_{i-1} , the sum of messages sent by this group is equal to $S = \sum_{s=1}^{r} \sum_{j=1}^{2k+1} m_{j,s} + \sum_{s=r+1}^{|g_{i-1}|} m_s = \sum_{s=1}^{r} [(k+1)v_s + k(-v_s)] + \sum_{s=r+1}^{|g_{i-1}|} v_s = \sum_{p \in g_{i-1}} v_p$. Where p is a participant node, r is the number of participants within the group g_{i-1} using $2k+1$ messages to express their votes and $m_{j,s}$ is thej^{th} message sent by the participant s. In the second step, members of the group g_i will receive a set of messages from the group g_{i-1}. The computed local tally of the group g_i is equal to $LT = \sum_{p' \in g_i} \sum_{p \in g_{i-1}} m_p^{p'}$, where $m_p^{p'}$ is a message sent from participant p to p'. Since we assume only honest nodes, the set of messages sent by nodes in g_{i-1} is the set of ballots received in g_i. Therefore, the local tally computed in the given group g_i reflects the vote of all nodes in g_{i-1}.

During the last step, each participant will forward local tallies received along the ring. The global tally is computed by each participant once all local tallies are received (i.e. the number of local tallies is equal to the number of groups). Since participants do honestly forward the local tallies along the ring and the messages are eventually received, each node ends up with the correct values for the local tallies of every group, thus the correct global tally $GT = \sum_{g=0}^{G-1} LT_g = \sum_{p \in \{N_1\}} v_p + \sum_{p \in \{N_{2k+1}\}} v_p$, where $\{N_1\}$ is the set of participants using a unique message and $\{N_{2k+1}\}$ represents the set of participants using $2k+1$ messages to express their votes.

Proof: (Termination). In the proposed protocol each participant knows the number of its proxies, its clients and its officemates. So nodes know the number of messages they are supposed to receive in each step. Since every participant node sends the required number of messages and every message eventually arrives, each step completes. As the algorithm is a finite sequence of steps (i.e. 3 steps), it is guaranteed to eventually terminate.

B. PDP's Complexity

In order to compare the complexity of DPol, the distributed polling protocol proposed in [1], and PDP, we have measured the average of the number of messages sent by a participant during different phases of the protocol, and the average time spent to execute the algorithm. Let us note that the performance measurements are made based on an experimentation of the PDP protocol.

In DPol, all participants share their votes into $2k+1$ messages. The asymptotic average number of messages sent by a node in group g_i is $O(G.k + |g_i|)$ [1]. Since in PDP only a subset of participants will share its votes into $2k+1$ messages, we predict that DPol will be r times more expensive than PDP in terms of message complexity, where r is the proportion of nodes voting with a single message.

The protocols were launched several times, varying the number of nodes and the number of groups. Figures [3 - 6] show the impact of the number of nodes on the number of exchanged messages and voting duration. The number of nodes varies from 50 to 400. It shows that PDP requires fewer messages than DPol and the execution time is lower compared to DPol's voting duration even if they have the same number of steps. Figure 3 shows that the number of messages increases linearly with the number of nodes for both protocols, but PDP scales better than DPol. In figure 4, the difference in the number of messages between DPol and PDP becomes increasingly wide, which is due to the fact

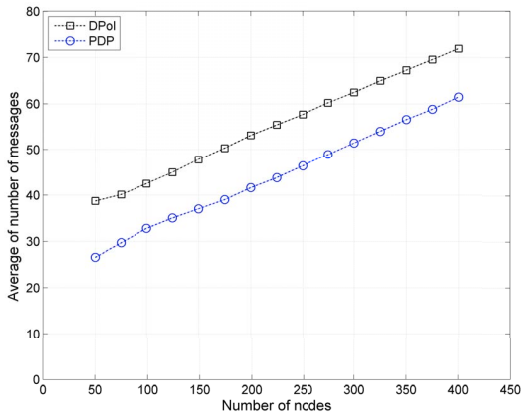

Figure 3. Number of messages vs. Number of nodes (10 groups)

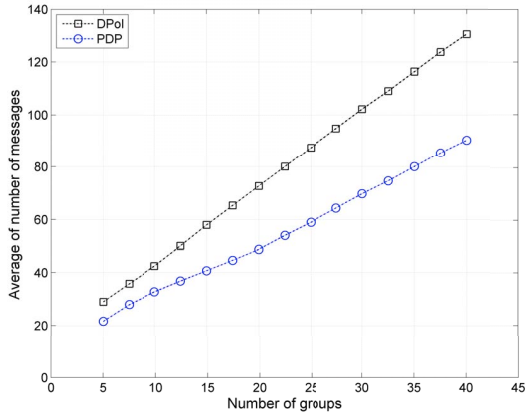

Figure 4. Number of messages vs. Number of groups (10 nodes per group)

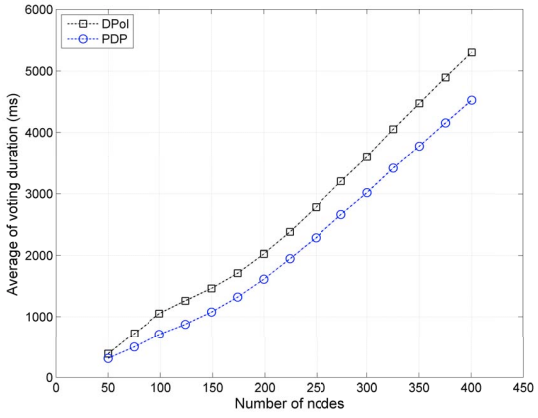

Figure 5. Voting duration vs. Number of nodes (10 groups)

that increasing the number of groups leads to an increase in the number of nodes.

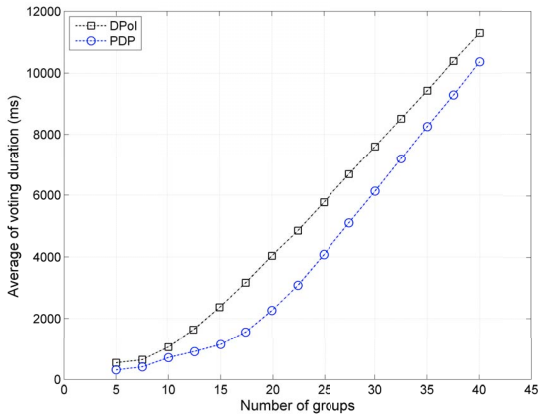

Figure 6. Voting duration vs. Number of groups (10 nodes per group)

The aim of figures 5 and 6 consists in studying the impact of the number of nodes and groups on voting duration for DPol and PDP. Figures 5 and 6 show the gain in terms of duration. The figures depict that the necessary time for the voting session increases with the increase of number of groups and nodes.

VI. Privacy Analysis

A. Reliable Detection

In this section, we will discuss the case when the vote of a given participant may be disclosed. Let us recall that participants express their votes either in the form of $2k+1$ messages or by sending only a single message. There are two cases where the vote of a participant can be surely disclosed. The first case is when $k+1$ dishonest proxies received $k+1$ messages containing the most represented value. The second case consists in participants using only 1 message to express their votes. In order to know the type of a given message, at least $N-2k+1$ dishonest proxies must collude, where N is the number of participants in a group. Therefore, if one of them receives the message of a participant using a single share, the vote of the sender will be certainly disclosed. Since the nodes are uniformly distributed, we can estimate the upper bound of the number of dishonest nodes in a group. Let us note B the maximum number of dishonest participants in a group.

Lemma 1: The probability, for a given participant using $2k+1$ messages, to have its vote recovered by a coalition of $B \geq k+1$ dishonest nodes is bounded by $(\frac{r}{N})(\frac{B}{N})^{k+1}$.

Proof 1: Participants using $2k+1$ messages to express their votes, send $k+1$ messages with a given value (i.e. -1 or $+1$). The vote of such participant is surely revealed by a coalition of dishonest participants if and only if $k+1$ proxies that received the $k+1$ shares containing the most represented value collude. In PDP, only r participants split their votes,

so the probability that the vote of these participants to be disclosed is $(\frac{r}{N})[\binom{B}{k+1}/\binom{N}{k+1}] \leq (\frac{r}{N})(\frac{B}{N})^{k+1}$.

Lemma 2: The probability, for a given participant using a single message, to have its vote recovered by a coalition of $B \geq N - 2k + 1$ dishonest nodes is $(\frac{B}{N})(\frac{N-r}{N})$.

Proof 2: The vote of a participant, using only a single message to express its vote, can be disclosed if and only if $N - 2k + 1$ dishonest proxies collude and one of these proxies receives the vote of the victim. This is due to the fact that the messages do not contain any information about their types either being a single or a shared message (i.e. the sender has split its vote or not). Thus, the probability that the vote of a participant using a single message to be surely disclosed is $(\frac{B}{N})(\frac{N-r}{N})$, where r is the number of participants sharing their votes.

Theorem: When at least $2k-1$ participants are honest, the probability for a given participant to have its vote recovered with certainty by a coalition of B dishonest nodes is bounded by $(\frac{r}{N})(\frac{B}{N})^{k+1}$.

Proof: The vote of a participant using $2k + 1$ messages can be disclosed only if $k+1$ proxies that received the most represented value collude. Accordingly, if the number of dishonest nodes B is less than $k+1$, the vote of a participant using $2k + 1$ messages cannot be surely revealed even if all dishonest nodes collude. However, if they are more than $k + 1$ dishonest nodes, the probability that the vote can be disclosed is bounded by the value $(\frac{r}{N})(\frac{B}{N})^{k+1}$ (Lemma 1). On the other hand, the votes of participants using single messages require at least $N - 2k + 1$ colluding dishonest nodes to be disclosed with certainty, and the probability of a certain detection is equal to $(\frac{B}{N})(\frac{N-r}{N})$ (Lemma 2). Summing the conditional probabilities (Lemmas 1 and 2) bounds, in the worst case (i.e. where $B \geq N - 2k + 1$), the probability of a certain detection by : $(\frac{r}{N})(\frac{B}{N})^{k+1} + (\frac{N-r}{N})(\frac{B}{N})$. However, saying that $B \geq N - 2k + 1$ means that less than $2k-1$ participants are honest. This condition is rarely true. Therefore, the probability in general is bounded by $(\frac{r}{N})(\frac{B}{N})^{k+1}$.

Privacy comparison

In order to compare the privacy in DPol with the privacy ensured by PDP, we have traced the curves that represent the probability that the vote of a given participant will be certainly discovered. Figure 7 displays the variation of this probability according to the number of dishonest nodes.

In this study, we consider a group of 100 participants with $k = 1$ (i.e. participants using $2k + 1$ messages will send 3 messages to express their votes). r is the number of participants using $2k + 1$ shares, this value changes from one curve to another in figure 7. In the x-axis, we have the number of dishonest participants and the y-axis represents the probability that the vote of a participant is certainly disclosed.

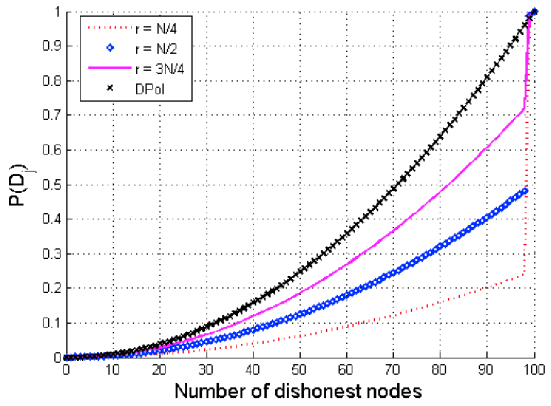

Figure 7. Privacy in PDP and DPol

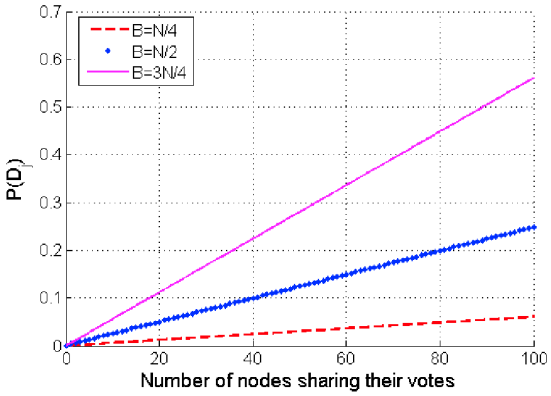

Figure 8. Impact of participant sharing their votes on the probability of reliable detection

As depicted in figure 7, PDP in all its variants offers more privacy than DPol as long as the number of dishonest participants does not exceed $N - 2k + 1$. In addition, when the number of participants using $2k+1$ messages decreases, the privacy increases. Thus, we get the maximum privacy of PDP, based on reliable detection, when just one participant shares its vote and the number of dishonest participants is less than $N - 2k + 1$.

Figure 8 represents the impact of the parameter r (i.e. the number of participants expressing their votes with $2k + 1$ messages) on the probability of reliable detection. From this figure, we can easily conclude that the privacy decreases with the increase of the parameter r.

These results are due to the fact that, dishonest nodes are willing to be sure about the disclosed participants' votes. If dishonest nodes receive a single message from an honest participant, they have to verify that this participant has a single proxy. This verification cannot be done if they are less than $N - 2k + 1$ nodes. Thereby, detecting votes of participants using $2k + 1$ messages is easier than detecting

votes of participants using single messages since $k+1$ dishonest nodes suffice. For this reason, when the number of participants sharing their votes increases, the probability of the certain detection increases.

Let us recall that the current analysis consists in reliable detection of honest participant's vote. This means that dishonest nodes will be sure that the revealed value represents the real vote. In the next subsection, we focus on the uncertain detection where participants are not sure of the certainty of the detected vote.

B. Uncertain Detection

The aim of the subsection is to study the uncertain detection of an honest participant's vote. The uncertain detection consists in detecting votes without sureness. This happens when a set of dishonest participants agree on the following rule: {upon reception of a single message, it will be considered as the sender's vote}. This means that, when a set of dishonest proxies receives a single message from an honest client, they will suppose that the received message is the client's vote. When their decision is correct, they succeed to discover the vote. However, they fail if the node has sent $2k+1$ messages to express its vote, and the set of dishonest nodes received only a message which does not represent the participant's vote. Dishonest participants will succeed to disclose the vote if one of the following events occurs:

- The honest participant has sent only a single message and it was received by dishonest proxies.
- The second case is when the following conditions are satisfied:
  - The client has sent $2k+1$ messages to express his vote.
  - The set of dishonest proxies received only one message representing the participant's vote.
  - The other k messages and their complementary were received by honest participants.

The probability to discover the vote of an honest participant is $P = P(D_j/_{j\in\{N_{2k+1}\}}).P(j \in \{N_{2k+1}\}) + P(D_j/_{j\in\{N_1\}}).P(j \in \{N_1\})$ where $\{N_1\}$ represents the set of participants voting with single messages and $\{N_{2k+1}\}$ is the set of participants voting with $2k+1$ messages. While r is the number of participants sharing their votes, $P(j \in \{N_{2k+1}\}) = \frac{r}{N}$ and $P(j \in \{N_1\}) = \frac{N-r}{N}$. It is clear that the probability of the first event is equal to $P(D_j/_{j\in\{N_1\}}) = \frac{B}{N}$. The probability of the second event is expressed after reduction by the formula: $P(D_j/_{j\in\{N_{2k+1}\}}) = [\binom{B}{1}\binom{N-B}{k}\binom{N-B-k}{k}/\binom{N}{k+1}\binom{N-k-1}{k}]$. By summing the conditional probabilities, we can conclude that the probability of the uncertain detection may be expressed as follows: $P = \frac{B}{N}[\frac{N-r}{N} + r(k+1)\prod_{i=0}^{B-1}\frac{N-2k-i}{N-i}]$.

Figure 9 shows the impact of dishonest nodes on the probability of the uncertain detection. As depicted in this figure, the probability of uncertain detection increases with the decrease of the number of participants sharing their votes. Therefore, the reliable detection of participants' votes decreases if the number of participants sharing their votes decreases. But, this is offset by the increasing uncertain detection. This suggests that the parameter r that represents the number of nodes splitting their votes should be chosen according to the strategy of dishonest participants. If we assume that dishonest participants will try to get any information about participants' votes, we should augment the value of r resulting in a decreasing probability of uncertain detection. Otherwise, we should choose a smaller value of r to decrease the probability of the reliable detection.

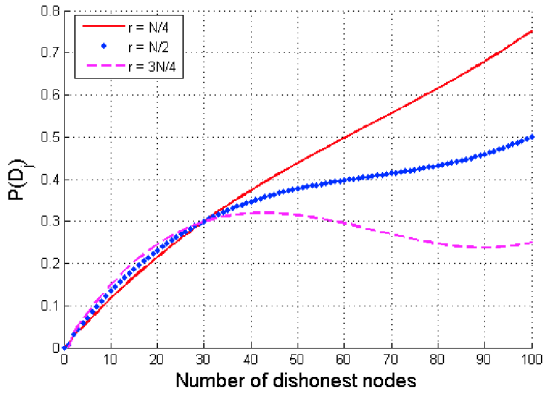

Figure 9. Uncertain detection

C. Combining Reliable and Uncertain Detections

In this subsection, we consider dishonest nodes trying to disclose an honest participant vote either being sure of the detected value or not. The dishonest participant will firstly try to detect certainly the participant's vote (respecting reliable detection rules). If they did not succeed to disclose the vote with certainty, they will use the uncertain detection (based on uncertain detection rule).

The probability of detecting a participant vote, considering such adversary act, is equal to the maximum of certain and uncertain detection probabilities.

Figure 10 shows the probability of detection with an increasing number of nodes sharing their votes. We consider the case where a group of 100 participants uses either 1 or 3 messages to express their votes (i.e. $k=1$), and $B = 3N/4$ are dishonest. As depicted in the figure, in a system of $N = 100$ nodes where $3N/4$ are dishonest. We get the best privacy when $r = 62$ participants split their votes into 3 messages, leading to a maximum probability of detection equals to 0.3476.

VII. Malicious Act Analysis

After studying different possibilities to reveal participants' votes, the aim of this section is to look into the impact of dishonest nodes on the final outcome of the protocol.

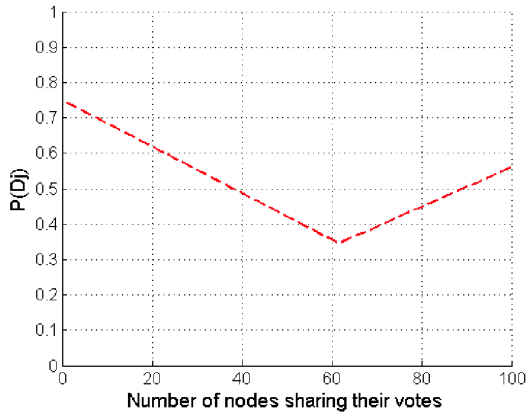

Figure 10. Maximum of the probabilities of detection

Dishonest participants may lead different types of attacks. These attacks aim at biasing the global outcome.

Participants splitting their votes may affect the global outcome without being detected with probability 1 by one of the following attacks:

- They can send more than $k+1$ messages with the same value. This attack may occur during the first phase of the protocol (i.e. the sharing phase).
- During the computation of the individual tally, a dishonest participant may invert the content of some messages.

A participant expressing its vote with a single message may affect the global outcome without being detected with probability 1 only during the computation of the individual tally. As a dishonest node receives $|P_c|$ messages, in the worst case, the node receives N messages, thus the maximum impact is $2N$.

As discussed above, participants sending $2k+1$ messages to express their votes may affect the global tally by two ways. The first way consists in sending more than $k+1$ messages with the same value (i.e. $k+5$ +1-messages, $k-4$ -1-messages). In the worst case, the client sends $2k+1$ messages with the same value leading to a maximum impact on the outcome equals to $2k$. The second kind of dishonest acts occurs during the individual tally computation where the impact on the output is bounded by $2N$. This leads to a global impact of $2(k+N)$ for participants sharing their votes.

Assuming that α dishonest nodes share their votes, the impact of the set of dishonest participants on the global outcome is bounded by $2(k\alpha + BN)$.

Therefore, the number of nodes splitting their votes has an impact on the protocol complexity, the ensured privacy, and on the result's accuracy due to dishonest act. If the votes require higher privacy, we can assume the uncertain detection. Thus, we should advance the number of participants sharing their votes to enhance the privacy. Consequently, the complexity and the impact on the outcome will increase. Otherwise, if we care only of the certain detection of participants' votes, the number of nodes splitting their votes should be decreased in order to reduce the number of exchanged messages and the impact on the global outcome.

VIII. Conclusion

This paper presents PDP, a probabilistic Private Distributed Polling protocol that allows a set of users to conduct a poll without revealing their inputs. The polling result is computed by participants themselves without interaction with a third entity, be it trusted or not. PDP does not rely on heavyweight cryptographic techniques. We proved that data privacy preserved by the protocol is ensured with certain probability and it is related to a decision rule used by dishonest participants. PDP ensures privacy such that a participant input is only known with certainty to its owner when the number of dishonest participants is not over a threshold. In the absence of data loss or malicious acts, the polling protocol is accurate. We proved also that the impact of dishonest participants on the global outcome is bounded by $2(k\alpha + BN)$. As future work, we plan to enhance the robustness of the protocol and we plan to study its behavior under node departure.

Acknowledgment

The authors are very grateful to Djalel Meskaldji for fruitful comments and suggestions about the privacy analysis.

References

[1] R. Guerraoui, K. Huguenin, A. M. Kermarrec, M. Monod and Y. Vigfusson, *Decentralized Polling With Respectable Participants*, in Journal of Parallel and Distributed Computing, Elsevier, vol. 72, pp. 13-26, 2012.

[2] W. Du, and M.J. Atallah, *Secure multi-party computation problems and their applications: a review and open problems*, in Proceedings of the 2001 workshop on New security paradigms (NSPW'01), Cloudcroft, New Mexico, September 11-13, pp.13-22, 2001.

[3] A.C. Yao, *Protocols for secure computations*, in Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS'82), Chicago, Illinois, USA, November 03-05, pp. 160-164, 1982.

[4] B. Su, and T. Wang, *Design and analysis for private determination protocol of segment-circle position relation*, in Proceedings of the Industrial Control and Electronics Engineering (ICICEE'12), August 23-25, pp. 1430-1433, 2012.

[5] D. Chaum, *Untraceable electronic mail, return addresses, and digital pseudonyms*, in Communications of the ACM, vol. 24, pp. 84-90, 1981.

[6] D. Chaum, *Security without identification: transaction systems to make big brother obsolete*, in Communications of the ACM, vol. 28, pp. 1030-1044, 1985.

[7] T. Elgamal, *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*, in the IEEE Transactions on Information Theory, vol. 31, pp 469-472, 1985.

[8] P. Paillier, *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*, in Proceedings of the 17th annual international conference on Theory and application of cryptographic techniques (EUROCRYPT'99), Prague, Czech Republic, May 2-6, pp. 223-238, 1999.

[9] C. Park, K. Itoh, and K. Kurosawa, *Efficient Anonymous Channel and All/Nothing Election Scheme*, in Proceedings of the 12th annual international conference on Theory and application of cryptographic techniques (EUROCRYPT93), Lofthus, Norway, May 23-27, pp 248-259, 1993.

[10] M. Jackobson, A. Juels, and R.L. Rivest, *Making Mix Nets Robust for Electronic Voting by Randomized Partial Checking*, in Proceedings of the 11th USENIX Security Symposium (Sec'02), August 5-9, pp. 339-353, 2002.

[11] D. Bruschi, I. N. Fovino and A. Lanzi, *A Protocol for Anonymous and Accurate E-Polling*, in Proceedings of the 2005 international conference on E-Government: towards Electronic Democracy (TCGOV'05), Bolzano, Italy, March 2-4, pp. 112-121, 2005.

[12] R. Cramer, R. Gennaro and B. Schoenmakers, *A secure and optimally efficient multi-authority election scheme*, in Proceedings of the 16th annual international conference on Theory and application of cryptographic techniques (EUROCRYPT'97), Konstanz, Germany, May 11-15, pp. 103-118, 1997.

[13] M. Hirt and K. Sako, *Efficient receipt-free voting based on homomorphic encryption*, in Proceedings of the 19th international conference on Theory and application of cryptographic techniques (EUROCRYPT'00), Bruges, Belgium, May 14-18, pp. 539-556, 2000.

[14] I. Damgrd and M. Jurik, *A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System*, in Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography (PKC'01), Cheju Island, Korea, February 13-15, pp. 119-136, 2001.

[15] I. Damgrd, M. Jurik and J. B. Nielsen, *A generalization of Paillers public-key system with applications to electronic voting*, in the International Journal of Information Security - Special Issue on Special Purpose Protocols, vol. 9, pp. 371-385, 2010.

[16] O. Baudron, P. A. Fouque, D. Pointcheval, J. Stern and G. Poupard, *Practical multi-candidate election system*, In Proceedings of the 20th annual ACM symposium on Principles of distributed computing (PODC'01), Newport, RI, USA, August 26 - 28, pp. 274-283, 2001.

[17] K. Peng and F. Bao, *A Design of Secure Preferential E-Voting*, in Proceedings of the 2nd International Conference on E-Voting and Identity (VOTE-ID'09), Luxembourg, September 7-9, pp 141-156, 2009.

[18] A. Shamir, *How to share a secret*, in Communications of the ACM, vol. 22, pp. 612-613, 1999.

[19] B. Chor, S. Goldwasser, S. Micali and B. Awerbuch, *Verifiable secret sharing and achieving simultaneity in the presence of faults*, in Proceedings of the 26th Annual Symposium on Foundations of Computer Science, (FOCS'85), Portland, Oregon, USA, October 21-23, pp. 383-395, 1985.

[20] J. C. Benaloh, *Secret sharing homomorphisms: keeping shares of a secret secret*, in Proceedings on Advances in cryptology (CRYPTO'86), Santa Barbara, California, USA, pp. 251-260, 1986.

[21] T. Rabin and M. Ben-Or, *Verifiable secret sharing and multiparty protocols with honest majority*, in Proceedings of the 21th annual ACM symposium on Theory of computing (STOC'89), Seattle, Washigton, USA, May 14-17, pp. 73-85, 1989.

[22] D. Malkhi and E. Pavlov, *Anonymity without 'Cryptography'*, in Proceedings of the 5th International Conference on Financial Cryptography (FC'01), Grand Cayman, British West Indies, February 19-22, pp. 117-135, 2002.

[23] D. Malkhi, O. Margo and E. Pavlov, *E-voting without 'Cryptography'*, in Proceedings of the 6th International Conference on Financial Cryptography, (FC'02), Southampton, Bermuda, March 11-14, pp. 1-15, 2002.