Information and Computation 234 (2014) 3–16

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Computing in social networks[☆], [☆☆]

Andrei Giurgiu[a], Rachid Guerraoui[a], Kévin Huguenin[a,*,1], Anne-Marie Kermarrec[b]

[a] EPFL, School of Computer and Communication Systems, EPFL, 1015 Lausanne, Switzerland
[b] INRIA Rennes – Bretagne Atlantique, Campus de Beaulieu, 35042 Rennes Cedex, France

ARTICLE INFO

Article history:
Available online 21 November 2013

Keywords:
Distributed computing
Security
Privacy
Social networks

ABSTRACT

This paper defines the problem of Scalable Secure computing in a Social network: we call it the S^3 problem. In short, nodes, directly reflecting on associated users, need to compute a symmetric function $f : V^n \to U$ of their inputs in a set of constant size, in a *scalable* and *secure* way. Scalability means that the spatial, computational and message complexity of the distributed computation does not grow too fast with the number of nodes n. Security encompasses (1) accuracy and (2) privacy: accuracy holds when the distance from the output to the ideal result is negligible with respect to the maximum distance between any two possible results; privacy is characterized by how the information disclosed by the computation helps faulty nodes infer inputs of non-faulty nodes, which we capture in our context by the very notion of probabilistic anonymity.

We first prove that under mild regularity conditions the problem of computing an arbitrary function can be reduced to that of component-wise addition of vectors of integers. More specifically, if the function f is Lipschitz-continuous and the maximum distance between two possible results is $\Omega(n)$, any protocol that S^3-computes component-wise addition of vectors of integers S^3-computes f.

We then present AG-S3, a protocol that S^3-computes a class of aggregation functions, that is that can be expressed as a commutative monoid operation on U: $f(x_1, \ldots, x_n) = x_1 \oplus \cdots \oplus x_n$, assuming the number of faulty participants is at most $\sqrt{n}/\log^2 n$. We further prove that AG-S3 S^3-computes component-wise addition of vectors of integers thus extending its application spectrum to regular functions. Key to our protocol is a dedicated overlay structure that enables secret sharing and distributed verifications which leverage the social aspect of the network: nodes care about their reputation and do not want to be tagged as misbehaving.

1. Introduction

The past few years have witnessed an explosion of online social networks and the number of users of such networks is still growing by the day, *e.g.*, Facebook boasts by now more than 400 millions users. These networks constitute huge live

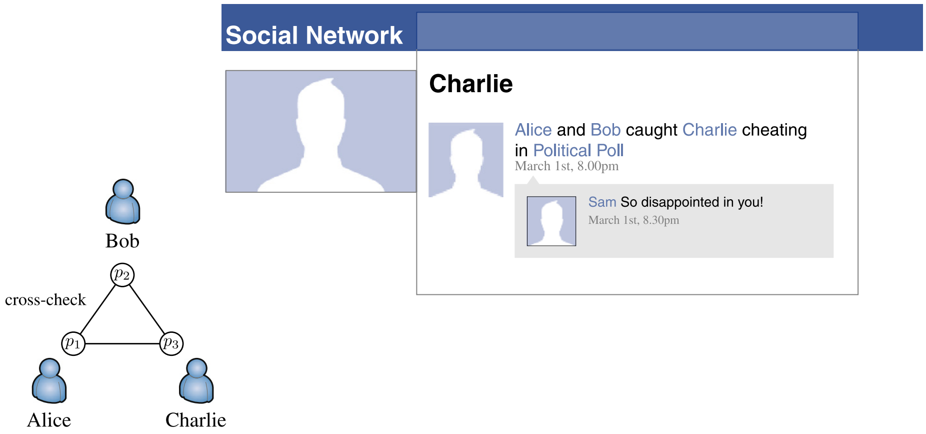
[☆] This paper is a revised and extended version of a paper that appeared in the Proceedings of the 12th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS 2010) [1].

[☆☆] This work has been partially supported by the ERC Starting Grant GOSSPLE number 204742.

[*] Corresponding author.
E-mail addresses: andrei.giurgiu@epfl.ch (A. Giurgiu), rachid.guerraoui@epfl.ch (R. Guerraoui), kevin.huguenin@epfl.ch (K. Huguenin), anne-marie.kermarrec@inria.fr (A.-M. Kermarrec).

[1] This research was partially carried out while Kévin Huguenin was working for his PhD at Université de Rennes 1/IRISA, France.

0890-5401/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ic.2013.11.001



Alice and Bob caught Charlie cheating in Political Poll
March 1st, 8.00pm

Sam So disappointed in you!
March 1st, 8.30pm

Fig. 1. Distributed verifications and reputation concern.

platforms that are exploited in many ways, from sharing personal information to conducting polls about political tendencies. An illustrative example of computation in a social network is crowdsourcing which exploits human-based computation capabilities and subjective and personal information of a group of users as a source of knowledge or ideas, *e.g.*, Amazon Mechanical Turk. It is clearly appealing to perform large-scale general purpose computations on such platforms and one might be tempted to use a central authority for that, namely one provided by the company orchestrating the social network. Yet, this poses several privacy problems, besides scalability. For instance, there is no guarantee that Facebook will not make any commercial usage of the personal information of its users. In 2009, Facebook tried to change its privacy policy to impose new terms of use, granting the company a perpetual ownership of personal contents—even if the users decide to delete their account. The new policy was not adopted eventually, but highlighted the eagerness of such companies to use personal and sensitive information.

We argue for a decentralized approach where the participants in the social network keep their own data and perform computations in a distributed fashion without any central authority. A natural question that arises then is what distributed computations can be performed in such a decentralized setting. Our primary contribution is to lay the ground for expressing the question precisely. We refer to the underlying problem as the S^3 problem: *Scalable Secure computing in a Social network*. Whereas *scalability* characterizes the spatial, computational and message complexity of the computation, the *secure* aspect of S^3 encompasses accuracy and privacy. *Accuracy* refers to the robustness of the computation and aims at ensuring accurate results in the presence of dishonest participants. This is crucial in a distributed scheme where dishonest participants might, besides disrupting their own input, also disrupt any intermediary result for which they are responsible. The main challenge is to limit the amount of bias caused by dishonest participants. *Privacy* is characterized by the amount of information on the inputs disclosed to other nodes by the computation. Intuitively, achieving all three requirements seems impossible. Clearly, tolerating dishonest players and ensuring privacy calls for cryptographic primitives. Yet, cryptographic schemes, typically used for multi-party computations, involve too high a computation overhead and rely on higher mathematics and the intractability of certain computations [2–4]. Instead, we leverage users' concern for reputation using an information theoretical approach and alleviate the need for cryptographic primitives. A characteristic of the social network context is indeed that the nodes are in fact users who might not want to reveal their input, nor expose their misbehavior if any. This reputation concern, as illustrated in Fig. 1, determines the extent to which dishonest nodes act: up to the point where their misbehavior remains discrete enough not to be discovered. In a system where users report on the misbehaviors they detect, dishonest node might be tempted to issue spurious reports on other users. However, in the context of social networks, two key factors help thwarting such a threat: First, reports are intended to be read by users (not programs) who can assess the credibility of the reports and decide whether to take them into account; Second, the knowledge of the social ties between users can be leveraged. For instance, reports from an enemy or a joint report issued by users that are connected in the social network could be disregarded. Such techniques proved efficient in areas as diverse as on-line games [5], recommendation systems [6], and spam filtering [7].

Solving the S^3 problem is challenging, despite leveraging this reputation concern: to ensure privacy, an algorithm must ensure that the information obtained by the coalition of faulty nodes during the protocol is not enough to determine with certainty a node's input. This property should hold even when all the non-faulty nodes except one have the same inputs: faulty nodes taking part in the computation must not know which non-faulty node had a different input. This requires the existence of two configurations of inputs that differ for two non-faulty nodes having different inputs, which with high probability lead to the same sequence of messages received by the faulty nodes. In turn, this comes down to *swapping* two nodes' inputs transparently (from the standpoint of the faulty nodes), which is challenging when the protocol needs to be also scalable and accurate. The scalability requirement (*i.e.*, each node communicates with a limited number of nodes) makes it difficult to find a chain of messages that can be swapped transparently between any two nodes in the system. The trade-off between privacy and accuracy can be illustrated by the following paradox: on the one hand verifying that nodes