Lightweight Privacy-Preserving Averaging for the Internet of Things

Tristan Allard
Université de Rennes 1, IRISA
tristan.allard@irisa.fr

Davide Frey
Inria Rennes
davide.frey@inria.fr

George Giakkoupis
Inria Rennes
george.giakkoupis@inria.fr

Julien Lepiller
Université de Rennes 1, IRISA
julien.lepiller@irisa.fr

Abstract

The number of connected devices is growing continuously, and so is their presence into our everyday lives. From GPS-enabled fitness trackers, to smart fridges that tell us what we need to buy at the grocery store, connected devices—things—have the potential to collect and make available significant amounts of information. On the one hand, this information may provide useful services to users, and constitute a statistical gold mine. On the other, its availability poses serious privacy threats for users. In this paper we propose a novel protocol that makes it possible to aggregate personal information collected by smart devices in the form of an average, while preventing attackers from learning the details of the non-aggregated data.[1]

1 Introduction

The Internet of Things has become a reality. Personal connected devices allow people to keep track of their weight, their fitness progress, or the distance they walk. Medical devices can inform doctors in real time about the health parameters of the patient wearing them. Smart environmental monitors make it possible to aggregate data into detailed maps to monitor pollution, weather, or other parameters. Home appliances like refrigerators connect to the Internet to tell people what to buy at the grocery store, while smart meters collect information from one's home electricity usage to compute area statistics and optimize distribution. It is easy to foresee that more and more devices will become available in the near future and that more and more of them will access and make available personal data.

These data constitute a gold mine not only for their direct recipients—doctors, users, electrical companies—but also for researchers, companies, or even end users interested in global statistical information. Electrical companies already use data about electric consumption to manage their smart-grid infrastructures. Public health agency could use data from health sensors to identify areas of a city that have a higher risk for a given disease. Supermarkets may use the data from smart fridges to adapt their stocks to the demands of their customers. However, the state-of-the art equates the implementation of these use cases with clear threats to the privacy of users.

Today's typical IoT applications collect the data of huge numbers of users into a single data store, possibly on the cloud or on a private data center, and process it, for example to compute relevant statistics. But this gives the application provider unconditioned

[1] Albeit appearing last in alphabetical order, Julien Lepiller was the main contributor to the work in this paper.

access to the data of all users.

A promising solution to the privacy issues associated with the analysis of large-scale privacy-sensitive information consists instead in decentralizing the processing itself. Keeping data where it is generated prevents service providers from accessing sensitive user information. However, decentralizing computation, whether on the edge or through completely peer-to-peer protocols, does not automatically eliminate all privacy risks. Even if sensors themselves aggregate data in a peer-to-peer fashion, they may still leak sensitive information to potentially malicious neighboring devices.

To address privacy in decentralized data processing, some authors have proposed the use of homomorphic encryption techniques [1, 5]. But like in the case of centralized solutions, the high cost associated with encryption and decryption operations makes them ill-suited to environments consisting of low-power devices like most IoT nodes.

In this paper, we address the limitations of existing techniques by proposing a novel lightweight protocol for decentralized averaging. The protocol exploits randomness and decomposition into shares as techniques to obfuscate the value associated with each node, and lightweight encryption techniques to withstand eavesdropping attacks. Although a thorough evaluation is out of the scope of this paper, we briefly analyze the protocol's resistance to colluding nodes and to eavesdropping attacks.

2 Background and Requirements

We focus on the problem of computing the average of a set of values in a decentralized manner while maintaining the individual values private. To this end, we take inspiration from existing decentralized averaging protocols, and in particular from the gossip-based aggregation solution in [7].

In the epidemic protocol in [7], each participating node maintains and iteratively refines a local approximation of the desired aggregate (the average). Initially, each node sets the value of its local approximation to its own value. Then it updates it as a result of gossip exchanges. In particular, each node, u, periodically contacts a random other node, v. The two nodes, u and v, exchange and update their values so that they both correspond to the average of the two values before the exchange.

This simple protocol causes the local estimates of all nodes to converge to the average of all the initial values. [7] further analyzes the speed of convergence and shows that it does not depend on the number of nodes but only on the variance of the initial distribution of values. This makes the protocol particularly suited to very large scale networks, and the very simple computation performed at every gossip exchange makes the protocol an excellent candidate for monitoring devices with limited computing power.

Unfortunately, this decentralized averaging protocol requires each node to exchange its initial value with another node at the beginning of the protocol. This makes the protocol unable to operate on privacy-sensitive information. To address this limitation, we propose a protocol that is at the same time: (i) lightweight, and (ii) resistant to honest-but-curious adversaries.

2.1 Lightweight Operation

Large-scale systems consisting of communicating "things" require lightweight protocols that can operate on devices with limited memory, computation, and communication capabilities and rules out solutions based on expensive encryption techniques. For example, Chiaroscuro [1] includes an privacy-preserving variant of [7]. But we experimentally verified that its encryption and decryption operations can take as long as 200s on a RaspberryPI, an already fairly powerful device.

2.2 Attack Resilience

We aim to design a protocol that can enable a set of cooperating users to compute the average of their initial values, while limiting the information that can be deduced by attackers. We consider two types of attackers: a node attacker, and an edge-attacker, which we describe in the following.

Node Attacker The node attacker models the case in which one or more users attempt to learn the initial values of a target user while participating in the protocol. We consider an honest-but-curious node attacker that controls a set of c nodes. Each of the controlled nodes, faithfully follows the protocol, but it forwards all the information it receives to the attacker which attempts to learn the initial value of a target node. Equivalently, we can view this attack as being played by a set of colluding users that follow the protocol, and exchange information to learn the initial value of the target user.

Edge/Eavesdropping Attacker The edge, or eavesdropping, attacker models instead the case in which a network service provider, or other external entity can observe all communication links between the users connected to its network. Again, we assume the attacker to be honest but curious, meaning that it cannot modify the data being exchanged. However, it will try to use all the information it observes to infer the initial value of a target node.

3 Private Lightweight Averaging through Random Shares

To address this problem, we start by observing that the protocol in [7] operates by having nodes exchange successive approximations of the average of all the nodes values. While towards the end of the protocol's execution, all values tend to be similar to each other, at the beginning they are very different, and at the first round, each node exchanges its own original value, the value that we wish to protect. More precisely, after k rounds of execution, we can intuitively observe that the value held and shared by a node will be the result of the average of $O(2^k)$ values. Based on this observation, it appears therefore clear that the protocol in [7] leaks most of the information about a node in the first few rounds of its execution, while it leaks almost none in later rounds.

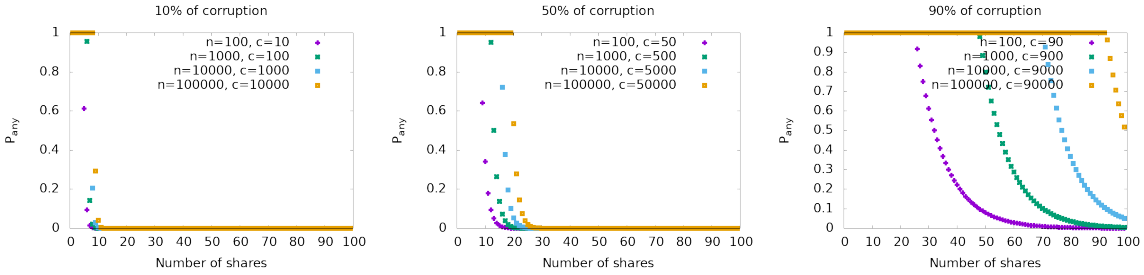
While some existing protocols protect this information by means of homomorphic encryption techniques [1], we adopt a more lightweight solution. We obfuscate the initial value of a node by decomposing it into a set of s *shares*, that is s values whose average corresponds to the initial value of the node. After creating the shares, each node creates a new virtual node for each of these shares and participates in the same protocol as in [7] with each of its virtual nodes. Specifically, in each round, a node u selects the value of one of its virtual nodes u_i and sends it to another node v; upon receiving the message, v also randomly selects one of its virtual nodes v_j and sends the corresponding value back to u. Both u and v then update the values of their corresponding shares so that they are both equal to $\frac{\mathcal{V}(u_i)+\mathcal{V}(v_j)}{2}$, where $\mathcal{V}(x)$ denotes the value of virtual node x.

Creating Random Shares This simple idea, which takes inspiration from well-known secret-sharing techniques [8], effectively protects the initial value of the target node from curious nodes (node attacker from Section 2.2) provided that the value of each virtual node lies sufficiently far from the node's real initial value.

To achieve this, each node builds its random shares by adding a random value to its original value. The choice of this value clearly determines the effectiveness of the obfuscation solution. For example, a value drawn from a bounded interval would make high and low values easy to guess. We therefore opt for a distribution over an infinite domain: a Gaussian with a mean of $\mu = 0$ and a variance of σ^2. We explore possible values for σ in Section 4.

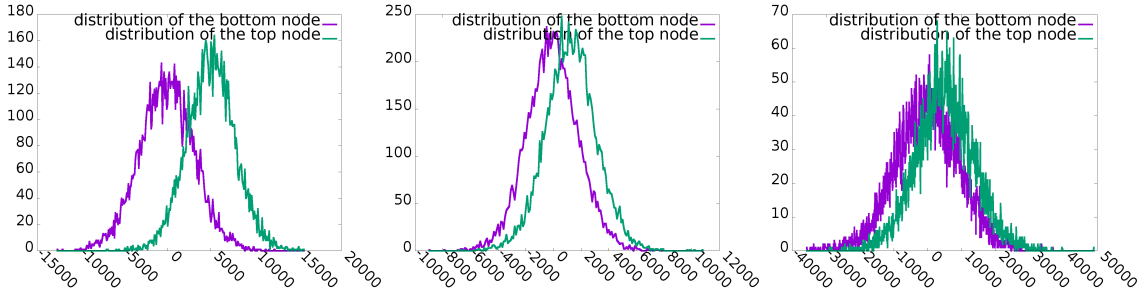
Protection from the Edge Attacker As suggested above, the use of virtual nodes with perturbed initial values serves as a protection from a node attacker that controls one or more curious network nodes. However, it offers absolutely no protection against an edge attacker, that is one that can observe all the messages exchanged by a node, or by a set of nodes. Such an attacker can simply monitor the messages sent and received by a node and average the corresponding values. By monitoring the exchanges, the attacker can thus easily compute the initial value of any node.

To counteract this kind of attack, we therefore aug-



(a) (b) (c)

Figure 1: Success probability for the node attacker.



(a) $k = 1, \sigma = 10,000$ (b) $k = 3, \sigma = 10,000$ (c) $k = 1, \sigma = 30,000$

Figure 2: Distributions seen by edge attacker: impact of the number of encrypted rounds, k, and of the noise's standard deviation, σ, with $s = 5$ shares.

ment our protocol with public-key encryption. To this end, we rely on the underlying peer-sampling protocol to disseminate the public key of nodes along the lines of [4], and use them to encrypt the messages sent during the first phases of the protocol. As we show in Section 4, a few encrypted rounds suffice to protect the privacy of users thanks to the convergence properties of the averaging protocol. Additionally, nodes send encrypted bait messages—empty messages—to random nodes to mask the set of nodes they are communicating with. Nodes receiving such messages reply with their own encrypted bait messages.

4 Analysis

Node Attacker We start by considering a simple attack in which a coalition of c colluding nodes, attempts to obtain all the shares from a given target node thereby learning its initial value.[2] In this preliminary analysis, we assume that nodes run a byzantine resilient peer-sampling protocol (e.g. [3]) that prevents nodes from communicating with the same node too often, and that the samples provided by such a protocol are independent of each other. Finally, we assume that nodes exchange the values of their shares in a round-robin fashion: they first do

[2] A more sophisticated attack could consist in acquiring only a subset of the shares thereby obtaining an approximation of the actual value. We leave the analysis of such a variant to future work.

a first exchange for all their shares, then a second exchange for all their shares and so on.

The most straightforward way for this attack to succeed consists in the target node's doing the first exchange for each of its shares with one of the colluding nodes. If the network comprises n nodes, of which c controlled by the attacker, then a node will exchange a given share with an attacker with a relatively small probability, $\frac{c}{n}$. However, the attacker may learn about the value of a node's share indirectly if it has information about the nodes that have previously communicated with the target. Precisely modeling the probability that an attacker may perform this kind of inference would be quite complicated. But we may easily compute an upper bound if we observe that if the target node communicates the initial value of a share to another node which is also communicating the initial value of its share, then the attacker will never be able to learn the exact value of either of these, but only their average. Conversely if the target node communicates the initial value of a share to a node that has already communicated its share value to someone else, then there is a possibility that the attacker may exploit this information. To model this probability, consider a node A that is exchanging its ith share with node B. Let $P(i,j)$ be the probability that B has already exchanged j values since the beginning of the protocol. If we consider i to be the average number of exchanges since the start of the protocol, then we can model $P(i,j)$ as a Poisson distribution with parameter i:

$$P(i,j) = \frac{i^j e^{-i}}{j!}.$$

The probability that, after i communication steps, a specific node has already exchanged all its original values P_{Xall} corresponds to the probability of having exchanged at least s values, that is:

$$P_{\mathsf{Xall}}(i) = 1 - \sum_{j=0}^{s-1} P(i,j).$$

The probability, P_{learn}, that the attacker will learn about all the s shares of a specific node can therefore be bounded from above as follows:

$$P_{\mathsf{learn}} \leq \prod_{i=1}^{s} \left(\frac{c}{n} + \frac{n-c}{n} \cdot P_{\mathsf{Xall}}(i) \right).$$

From this, we can bound the probability P_{any} of an attacker's learning all the shares of at least one node.

$$P_{\mathsf{any}} \leq (n-c) \cdot P_{\mathsf{learn}}.$$

Figure 1 depicts this probability and evaluates the number of shares required to manage different percentages of attackers (corruption) with increasing network sizes (n varying from 100 to 100,000). For example, the left plot shows that 10 shares provide enough if the attacker controls 10% of the nodes, while 30 are required to withstand an attacker controlling half of the network.

Edge Attacker The edge attacker has no way to acquire the initial values of a node's share since nodes encrypt the first rounds of exchanges. However, it can easily monitor all the data exchanged by nodes as soon as non-encrypted rounds begin. By monitoring the non-encrypted rounds, the attacker can easily estimate the average of a node's current share values. The information this provides will depend on the number of encrypted rounds as well as on the variance of the noise added when creating the shares. To evaluate this trade-off we ran an experiment with a network of $n = 1,000$ nodes. One *high-valued* node starts with an initial value of 10,000 while all the other *low-valued* nodes start with an initial value of 0. Initially each node creates its shares by adding a normally distributed noise $\mathcal{N}(val, \sigma)$. Then the protocol runs for k encrypted rounds after which we measure the average of the shares of the node that started with a value of 10,000, and that of a node that started with a value of 0. We repeat the experiment 10,000 times to obtain a distribution of values for both high- and low-valued nodes. Figure 2 compares the resulting distributions for several configurations. Results show that privacy can be enhanced by increasing either k or σ. Yet we expect that increasing σ may make the protocol more vulnerable to churn in the first rounds.

5 Related Work

The use of gossip to aggregate information lies on well established results [7]. But only recently have researchers started to explore how to tweak these well known aggregation protocols in order to preserve privacy. One of the authors of this paper proposed a decentralized system for k-means computation which provides a differentially private average [1]. Yet, the protocol relies on expensive homomorphic encryption primitives that make it unsuitable for IoT scenarios.

Another recent contribution [6] uses gossip to achieve differentially private stochastic gradient descent. The use of noise to achieve differential privacy may appear similar to what we do in this paper, but contrary to [6], we seek to eliminate noise to obtain as precise an average as possible, rather than a differentially private one. Some related work [5] proposes the use of shares for gradient descent learning. However, unlike our protocol, their solution still relies on expensive homomorphic encryption.

Finally, [2] proposes DiPA, a protocol that also relies on shares but with a more structured topology. This requires special mechanisms to deal with node failures which are not required in the case of gossip. Moreover DiPA does not take into account the presence of an edge attacker.

6 Conclusions

We presented a novel protocol for privacy-preserving averaging. The protocol is resilient to colluding nodes as well as to eavesdropping attacks, without relying on expensive homomorphic encryption operations. Our preliminary results encourage us to perform a more thorough evaluation of security and scalability properties, while addressing open questions like the impact of stronger attackers.

Acknowledgments. This work was partially funded by the Region of Brittany, France, by the French ANR project SocioPlug (ANR-13-INFR-0003), and by the DeSceNt project granted by the Labex CominLabs excellence laboratory (ANR-10-LABX-07-01).

References

[1] T. Allard, G. Hébrail, F. Masseglia, and E. Pacitti. Chiaroscuro: Transparency and privacy for massive personal time-series clustering. In *ACM SIGMOD*, pages 779–794, 2015.

[2] Y. Benkaouz and M. Erradi. A distributed protocol for privacy preserving aggregation with non-permanent participants. *Computing*, 97(9):893–912, 2015.

[3] E. Bortnikov, M. Gurevich, I. Keidar, G. Kliot, and A. Shraer. Brahms: Byzantine resilient random membership sampling. *Computer Networks*, 53:2340–2359, 2009.

[4] A. Boutet, D. Frey, A. Jégou, A.-M. Kermarrec, and H. B. Ribeiro. Freerec: An anonymous and distributed personalization architecture. *Computing*, 97(9):961–980, 2015.

[5] G. Danner and M. Jelasity. Fully distributed privacy preserving mini-batch gradient descent learning. In *DAIS. DisCoTec*, pages 30–44.

[6] I. Hegedus and M. Jelasity. Distributed differentially private stochastic gradient descent: An empirical study. In *Euromicro PDP2016*, pages 566–573, 2016.

[7] M. Jelasity, A. Montresor, and O. Babaoglu. Gossip-based aggregation in large dynamic networks. *TOCS*, 23(3):219–252, Aug. 2005.

[8] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, Nov. 1979.