Secure and Distributed Computations for a Personal Cloud

Riad Ladjel, Nicolas Anciaux, Philippe Pucheral, Guillaume Scerri

Secure and distributed computations for a personal cloud

Riad Ladjel[1,2] Nicolas Anciaux[1,2] Philippe Pucheral[2,1] Guillaume Scerri[2,1]
[1]Inria, France [2]U. Versailles St-Q., France
Fname.Lname@inria.fr Fname.Lname@uvsq.fr

CONTEXT

A large amount of economically valuable data is produced every day. Whether they come from smartphones, connected devices, sensors or smart meters, this data is a gold mine for the people holding it. This data is often stored in centralized servers, servers that usually belong to large corporations that collected it in the first place (Google, Amazon, Facebook, insurance companies etc.) The result of this situation is that users lose control over their own data. This represents a real threat to privacy, whether it is intentional (misuse, malicious attack), or just by negligence (data leakage, mismanagement). These threats point to the need for personal platforms which allow their users to collect, manage and share their own data. This is the essence of the self-data movement. Thanks to smart disclosure initiatives, users can access their personal data from the companies or government agencies that collected them. Concurrently, Personal Cloud solutions are flourishing. Their goal is to empower users to leverage their personal data for their own good.

MOTIVATION

While storing data in personal clouds increases user control over data, in the personal cloud context collaborative use of data is often overlooked. The benefits derived from exploiting data are considerable. A user may want to share her GPS position to have accurate traffic prediction [8], or her medical records to train a shared neural network so that it can detect several diseases [4, 10]. She may also want to adapt her energy contract based on her actual consumption without jeopardizing her privacy [9].A naive approach to this problem is to send personal data to a trusted third party who will perform said collaborative computations. This, however involves very strong trust in the third party's honesty. The goal of this work is to overcome this unrealistic trust assumption and propose a privacy preserving distributed computation framework for performing collaborative computations over a large number of personal clouds.

OBJECTIVE

Our objective is to propose a secure distributed computation protocol offering the same guaranties as secure multi-party computation (*SMC*)[3, 7] while keeping a profit-to-cost ratio as low as possible in case of an attack. In other words, it should be possible for parties to compute any function $f(x_1, x_2, ..., x_n)$ over their inputs while keeping them private. An individual cannot infer any information other than what she can infer from her own input and the final result. The result should be exact and not approximate. Unlike *SMC*, the proposed protocol should be scalable, with respect to the number of parties and the complexity of the computation.

LIMITS OF EXISTING SOLUTIONS

Secure Multi-party Computation (*SMC*)[3, 7] allows N users to perform a secure computation on their personal data. The users involved learn nothing more than what they can infer from their own input and the final result. These protocols are based on complex cryptographic techniques, which limit their scalability, in particular for very large number of parties. Ad-hoc *SMC* protocols have been proposed but these are not generic and do not allow all types of computation.
Differential privacy[5] Another technique to perform computations on personal data while respecting users' privacy is differential privacy. This consists in adding a quantity of noise to the result of a query before sending it to a third party. This approach has two main downsides: it typically requires a trusted third party to decide what noise to add, and, in order to have relevant results, it requires a large amount of data because of the added noise.
K-Anonymity [11] The principle of k-anonymity is to anonymize the data using different techniques (suppression, generalization ...). An individual in a k-anonymous database cannot be distinguished among k other individuals. This approach shares the same issues with Differential Privacy: it requires a central entity that anonymizes the data, and the anonymous data produced is not accurate thus greatly reduced its utility.
Privacy preserving schemes based on gossip protocols[1, 6] Gossip protocols are based on a communication technique that works the same way a rumor spreads. On every round each node randomly selects a node from its neighborhood to exchange information, after a sufficient number of rounds, the result of each node converges to the final result. Gossip protocols scale well but are not generic in terms of possible computations, which make them unsuitable for our case.

OUR APPROACH

We propose a protocol for secure distributed computations based on *Trusted Execution Environments (TEE)*. A *TEE* is a secure part of the processor, which guarantees confidentiality and integrity. Code executed inside a *TEE* is protected against all elements outside it, including the operating system. Several vendors propose their own Trusted Execution Environment : *Platform Security Processor* for *AMD* , *TrustZone* for ARM and *Software Guard Extensions* for *Intel*, the latter one will be used for the implementation and evaluation of our protocol. To be usable in our protocol a *TEE* must be able to do *Attested Computation*[1]. Our architecture is composed of a querier and several users who agree to contribute to the computation. They are all equipped with a *TEE*. The use of *TEE* increases the cost of an attack while distributing the computations over plenty of nodes decreases the benefits of attacking a single computation node.

PROBLEM STATEMENT

Basically, the use of *TEE* makes each computation node honest individually. This drastically changes traditional threat models. But it gives no guarantees on the distributed computation. The problem here is threefold : (1) we have to define a new threat model which fits well in our context, then based on this threat

[1] *"Attestation is the process of demonstrating that a piece of software has been properly instantiated on the platform. In Intel SGX it is the mechanism by which another party can gain confidence that the correct software is securely running within an enclave on an enabled platform"*[2]

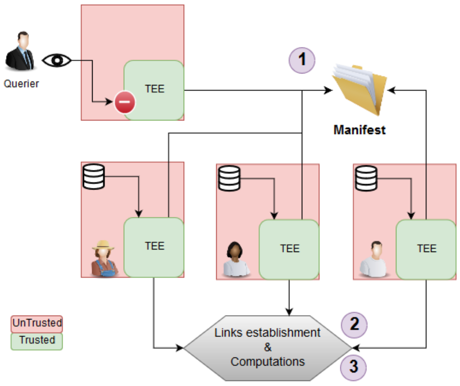

Figure 1: overview of the architecture

model, (2) we propose a protocol which forces computation nodes to follow a list of rules describing the distributed computation, to provide global guarantees on the correctness of the computation. But while *TEE* is supposed to offer confidentiality of its content, a small fraction of users could use side channel attacks to retrieve the content of their *TEE*. Thus, (3) we propose a set of countermeasures to reduce the benefit-to-cost ratio of an attack to the protocol.

Additionally, a successful attack by an individual or a set of colluding participants should neither compromise a large set of users' privacy nor be able to target a specific user. Moreover, the protocol should guarantee the integrity of the result with respect to the data provided by the participants.

TOWARDS A NEW THREAT MODEL

Traditional threat models do not fit well in a context involving *TEE*, therefore the first step in our work is to identify and propose a well adapted threat model. We consider two different ones (1) honest-but-curious users (participants and/or querier) who may try to infer some information without breaking the confidentiality of the *TEE* and (2) a small fraction of malicious users who break the confidentiality of their *TEE* by performing a side channel attack retrieving some information from it. Note that in the second case, the integrity guarantees of a *TEE* still hold.

SCALABLE AND GENERIC PROTOCOL

Our approach aims at designing a protocol that supports execution of any distributed computation, on a potentially large number of participants, while providing strong security guarantees. The protocol is organized in three phases (see Fig. 1). The first phase is to set up the environment; the querier broadcasts the function to be computed, together with the organization of the distributed computation. Users who want to participate to the computation register with the querier . When the number of participants is sufficient, the querier and participants collaborate to produce an execution table, which is a list of tasks to do for each participant (e.g. $user_1$ executes f_1 on $user_2$ data and sends her output to $user_3$). The list of participants, the topology and the execution table form a manifest. This manifest has to be public, so that everyone can verify it. Thanks to attestation, the manifest is generated with strong guarantees of integrity. In the second phase, participants establish links between them following the manifest. Users leverage attestation to prove that they are executing their assigned tasks from the manifest. Finally, the last phase is the actual computation and it depends of the algorithm implemented. This protocol provides strong integrity guarantees, as breaking integrity means compromising a tamper resistant *TEE*. Moreover, if a malicious participant performs a side channel attack, retrieving the content of its *TEE*, it only has access to a limited amount of data as prescribed by the manifest. As long as the distributed computation is organized in a sensible way, no party should have access to large amount of personal data, and the threat of large scale attacks is this mitigated. Additionally, note that all computations are done on clear data, the results produced are thus exact. Moreover, only simple cryptographic operations and bookkeeping inside *TEE* are required in addition to the actual computation. This allows for a very limited overhead with respect to performing the computation in the clear between mutually trusted parties.

SECURITY COUNTER-MEASURES

We are considering two ways to reduce the benefit-to-cost ratio (1) by proposing ad-hoc countermeasures, depending on the protocol and included in the computations. Let us take as a simple example the case of a group by computation. The computation can be broken down in many levels of partial aggregation in a tree like fashion where no one party get access to the raw data of all participants. At each level of aggregation, the participant only has access to a small subset of the partially data. This drastically reduces the benefit of an attack on a few computation nodes as no partial result holds a significant amount of precise information on a large number of participants.

(2) By proposing, protocol independent, general countermeasures. This kind of countermeasures can be implemented using the proposed protocol itself, as a pre processing step leading sample and/or pre-agregate or generalize the set of inputs data depending of the desired precision for the computation. For example in a group by aggregation on users' street, there is no need for a precise GPS localization.

ROADMAP

The next steps are to formalize the threat model and the protocol aiming at ultimately giving cryptographic proofs of the desired properties and finally validate this protocol on some examples; amongst other, training a shared neural network in a suitably decentralized way and a group by aggregation.

REFERENCES

[1] Tristan Allard, Davide Frey, George Giakkoupis, and Julien Lepiller. 2016. Lightweight privacy-preserving averaging for the internet of things. In *M4IOT*.

[2] Ittai Anati, Shay Gueron, Simon Johnson, and Vincent Scarlata. 2013. Innovative technology for CPU based attestation and sealing. In *HASP*.

[3] Ronald Cramer, Ivan Bjerre Damgård, and others. 2015. *Secure multiparty computation*. Cambridge University Press.

[4] Joseph A Cruz and David S Wishart. 2006. Applications of machine learning in cancer prediction and prognosis. *Cancer informatics* (2006).

[5] Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*.

[6] David Kempe, Alin Dobra, and Johannes Gehrke. 2003. Gossip-based computation of aggregate information. In *Foundations of Computer Science*.

[7] Yehuda Lindell and Benny Pinkas. 2009. Secure multiparty computation for privacy-preserving data mining. *Journal of Privacy and Confidentiality* (2009).

[8] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei-Yue Wang. 2015. Traffic flow prediction with big data: a deep learning approach. *IEEE Transactions on Intelligent Transportation Systems* (2015).

[9] Andrés Molina-Markham, Prashant Shenoy, Kevin Fu, Emmanuel Cecchet, and David Irwin. 2010. Private memoirs of a smart meter. In *Embedded sensing systems for energy-efficiency in building*.

[10] Sellappan Palaniappan and Rafiah Awang. 2008. Intelligent heart disease prediction system using data mining techniques. In *AICCSA*.

[11] Latanya Sweeney. 2002. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* (2002).