Secure Computations in Opportunistic Networks: An Edgelet Demonstration with a Medical Use-Case

Ludovic Javet
Inria Saclay Centre at Université Paris-Saclay
U. Versailles St-Quentin-en-Yvelines
Versailles, France
ludovic.javet@inria.fr

Nicolas Anciaux
Inria Saclay Centre at Université Paris-Saclay
U. Versailles St-Quentin-en-Yvelines
Versailles, France
nicolas.anciaux@inria.fr

Luc Bouganim
Inria Saclay Centre at Université Paris-Saclay
U. Versailles St-Quentin-en-Yvelines
Versailles, France
luc.bouganim@inria.fr

Léo Lamoureux
Inria Saclay Centre at Université Paris-Saclay
U. Versailles St-Quentin-en-Yvelines
Versailles, France
leo.salagnac-lamoureux@inria.fr

Philippe Pucheral
Inria Saclay Centre at Université Paris-Saclay
U. Versailles St-Quentin-en-Yvelines
Versailles, France
philippe.pucheral@inria.fr

***Abstract*—In this demonstration paper, we leverage the current convergence between Trusted Execution Environments and Opportunistic Networks to perform secure and privacy-preserving computations on personal devices. We call this convergence *Edgelet computing* and show that it can drastically change the way distributed processing over personal data is conceived. We demonstrate the pertinence of the approach through a real medical use-case being deployed in the field.**

I. Introduction

With the explosion in the number of connected devices – 29.3 billion by 2023 [1] –, the usual reliable, server-based and infrastructure-centric data management approach exhibits its limits in terms of efficiency, privacy, and energy consumption. In [2], we show that the recent convergence between Trusted Execution Environment (TEE) and Opportunistic Network (OppNet) can be leveraged to perform fully decentralized and secure computations among data scattered on multiple personal devices, without resorting to any central authority or infrastructure. Hence, powerful large-scale privacy preserving computations are within reach in a different – and more flexible – way than with homomorphic encryption [3], secure multiparty computation [4] or differential privacy techniques [5]. We call this approach *Edgelet computing*.

Edgelet computing paves the way for disruptive applications by (1) encouraging data sharing with strong privacy guarantees, (2) distributing liability for the processing among multiple devices/individuals rather than centralizing it on a single data processor (in the GDPR sense), and (3) leveraging the mobility of individuals and opportunistic communications among them to perform ad-hoc spatio-temporal queries. As a concrete example, we are today developing a medical use-case, on top of the DomYcile project [6]. DomYcile is a data management solution for elderly people receiving home care in the French Yvelines district. 8,000 patients are each being equipped with a secure home box (TEE-enabled device acting as an *edgelet*) where their medical and social records are stored. These boxes are not connected to the Internet for subscription cost, security, and acceptability reasons and are only accessible at the patient's home by healthcare workers. The DomYcile platform, set up by the Yvelines district, is open by design, so that third parties (e.g., patient associations, statistical agency, medical workers) can push new services of interest for the patients (e.g., querying ephemeral cohorts of consenting patients and delivering them afterwards valuable healthcare advices). In such context however, no one would be ready to endorse the responsibility of the global processing in case of compromise. Fig. 1 illustrates how the Edgelet computing paradigm can tackle this issue: the query is first approved by an authority (e.g., the CNIL - French regulatory agency), then broadcast by healthcare workers who implement the OppNet with encrypted messages between edgelets. Finally, a subset of secure boxes materializing the cohort act as *Data Contributors* while others are randomly selected to act as *Data Processors* that perform the computation.

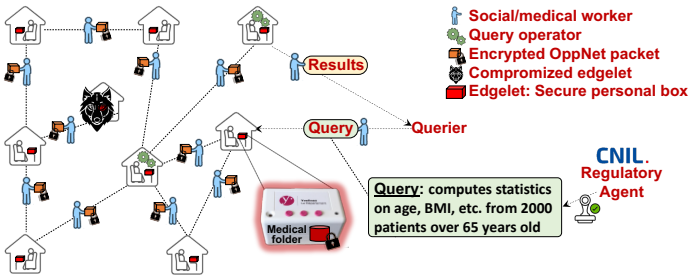

Fig. 1. Edgelet query in the DomYcile project

The main challenge is to design a scalable decentralized execution strategy that provides accurate results and limits the exposure of individual data in case of compromise, all within the OppNet context, prone to failures and message delays. In [2], we have proposed technical solutions to tackle this challenge and have discussed their impacts in terms of result consistency, performance and privacy. The objective of this demonstration is to present their practical interest through the use-case above. A video is available online (https://project.inria.fr/edgelet/).

II. Edgelet Computing Framework

Basic assumptions. On the technical side, Edgelet computing capitalizes on one hand on the security properties provided by TEEs [7] (integrity of the code executed on the devices and confidentiality of the manipulated data) and on the presence of TEEs in a fast-growing population of devices (from

smartphones to lightweight devices up to sensors). On the other hand, Edgelet computing exploits the asynchronous short-range communications (e.g., Wi-Fi or Bluetooth) and store-carry-and-forward message passing among devices provided by OppNets [8], eliminating the need to rely on a reliable infrastructure. Note that for simplicity, we consider epidemic message transmission and let optimized routing protocols exploiting user movement patterns for future work. On the functional side, Edgelet computing aims at performing complex processing tasks (e.g., machine learning, data mining, statistical queries) over rich data (e.g., healthcare folders, spending habits) hosted by edgelets seen as a horizontal partitioning of a globally shared database (i.e., each edgelet stores a partition of a database conforming to a common schema).

Threat model. Edgelet computing requires the introduction of an unusual threat model to capture the shift of responsibility from the Querier to the crowd (i.e., the participating edgelets). The Querier (e.g., the service provider) is said "*Ingenuous*" in the sense that it lacks the ability or willingness to set up a secure infrastructure. The edgelets owners (e.g., the patients) run on their side under the so-called "*wolf in sheepfold*" model: while each edgelet is equipped with a TEE, side-channel attacks compromising data confidentiality cannot be totally precluded, placing it in a "sealed glass" mode [9] (the integrity guarantee is preserved but not the confidentiality); these attacks being highly difficult to conduct – they require instrumenting the edgelet hardware –, we assume a large majority of honest participants (the lambs) and a few corrupted ones (the wolfs). Finally, a Trusted Regulatory Agent (e.g., the CNIL) certifies the query by reviewing and approving its code. The rest of the infrastructure is assumed to be untrusted, including the devices' operating system and the network. Under this threat model, the goal is to achieve a form of *crowd liability* where the liability of data processing is equally distributed among the participating edgelets. For each edgelet owner, the counterpart of its contribution to the crowd liability is to personally benefit from a processing that would not be rolled out otherwise (e.g., share common practices/knowledges among patients). Moreover, for a *lamb* user, the TEE and its embedded code actually endorse this liability on her behalf.

Execution plans. We represent computations by a Query Execution Plan (QEP), that is a directed graph where vertices materialize the operators to be computed and edges represent the dataflow among them, with messages sent through the OppNet. The simplest form of a QEP is a tree with Data Contributors at the leaves (one per edgelet contributing to the query with its data) sending the requested data to a *Snapshot Builder* operator, the role of which is to collect a representative snapshot of cardinality C (e.g., select tuples from C=2000 patients over 65 years old, as considered on Fig. 1). The Snapshot Builder then sends the representative snapshot to a *Computer* operator which, in turn, computes the final result and delivers it to the Querier. The Data Processors (here, the Snapshot Builder and the Computer) can be decomposed into sub-operators assigned to different edgelets for privacy or performance concerns. This can help minimizing the amount of data exposed at each edgelet by horizontally partitioning the dataset. This can also preclude the concomitant exposure, in the same edgelet, of data items that become sensitive when combined (e.g., a quasi-identifier) by vertically partitioning the dataset. This can finally help minimizing the workload (e.g., when energy consumption matters) or exploit the inherent Edgelet computing parallelism. A *Computing Combiner* operator must then be added at the root of the QEP to combine the outputs of all sub-operators.

Resiliency strategies. During a QEP execution, each edgelet acting as a Data Processor is a potential Single Point of Failure, with presumed failures encompassing temporary disconnection and excessive message delay caused by the OppNet. Fault tolerance is essential to make the approach applicable. We then explored in [2] two resiliency strategies, namely *Backup* and *Overcollection*, which differ on complexity and performance, on the accuracy of the final result and on the way privacy is enforced, making each strategy best adapted to different contexts. In this demonstration, we focus on the Overcollection strategy which is best adapted to our use-case where simplicity and performance take precedence and approximate results are acceptable (e.g., statistics, machine learning processes).

Overcollection sketched. Broadly speaking, the Overcollection principle works as follows. Let us consider a Computer operator that executes a distributive operator (e.g., a Sum) over a sample dataset. Instead of executing this Computer on a single edgelet, we distribute its execution over $n+m$ edgelets where each one processes a partition of the original dataset, with n the minimum number of partitions to be collected and m the overcollection parameter. The validity of the query is trivially preserved as long as less than m partitions are lost. This strategy requires that the commutativity rules between operators of a same QEP allow all distributive operators to be pushed down to the sub-QEPs and all non-distributive operators to be pushed up to the root. In practice, many data-intensive queries of interest satisfy this requirement. We show in [2] that general interest Machine Learning and Data Mining algorithms can also accommodate Overcollection despite their iterative nature and the fact that messages can be lost along the iterations. We also demonstrate that an appropriate implementation guarantees their convergence toward an accurate approximate result (with a principle similar to *Mini-batch K-Means* [10]), and this demonstration confirms its validity in practice.

III. Demonstration

Demonstration hardware platform. It includes a set of personal secure boxes, as deployed and delivered to each patient in the DomYcile project (see Fig. 2). Medical and social workers interact with the patient folder hosted in the box through a smartphone App. Secure boxes incorporate two STM32F417 microcontrollers (MCUs). The first MCU is dedicated to communications with the outside while the second manages the recorded data. This second MCU is connected to a µ-SD card hosting the patient's raw data, and to a tamper-proof TPM (Trusted Platform Module). This TPM secures the cryptographic keys and guarantees, during the secure boot, that the embedded code has not been tampered with. Hence, these secure boxes act as TEE enabled devices and play the role of edgelets.

Demonstration software platform. As shown in Fig 3, it consists of: (1) a GUI implemented in Dash Python that allows interactive configuration and visualization of Edgelet queries; (2) an Edgelet manager that orchestrates executions and communications between edgelets; (3) an OppNet modeler that

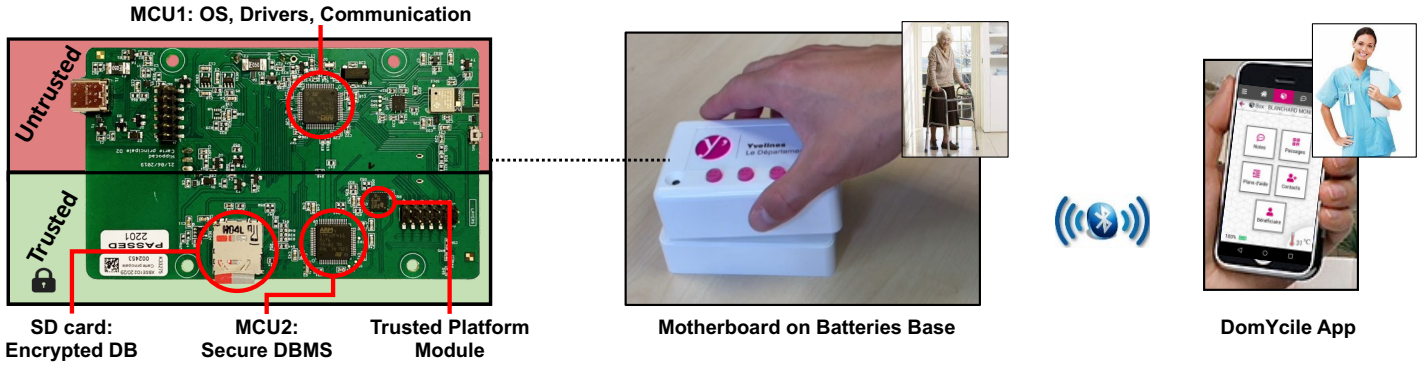

Fig. 2. Hardware of DomYcile secure boxes

models the massive distribution of edgelets over the city of Versailles (chief town of the Yvelines district). The OppNet modeler itself relies on the ONE simulator [11] and uses the generated message traces for the Edgelet manager to coordinate the executions in a similar way. Note that the Edgelet manager and OppNet modeler are only necessary for the demonstration and are not part of the platform deployed in the field.

Demonstration scenario. The attendees are first invited to select a query among two representative ones: (i) a *Grouping Sets* query [12] which allows multiple Group-By clauses to be evaluated within a single SQL query (to cross multiple statistics over a same cohort of patients); or (ii) a *K-Means* [10] followed by a Group By on the resulting clusters (to identify which characteristics most influence the dependency level of an elderly person). Then, following their intuition, the attendees can try to improve the privacy of the QEP of the selected query to reduce data exposure in case of TEEs compromise by adjusting the horizontal and vertical partitioning parameters presented in Section II. Next, we proceed to the real-time execution of the resulting QEP on the boxes available for the demonstration (concrete edgelets), the rest of the operators being associated to a configurable number of simulated edgelets to attest scalability. Finally, the attendees can change the ONE scenario (e.g., number of healthcare workers), impacting the failure presumption for each edgelet, and observe automatic changes in the QEP (overcollection ratio) to keep it resilient. The demo allows attendees to visualize, step by step, the query execution.

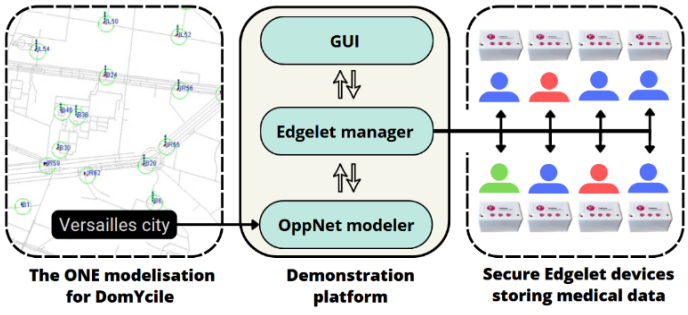

Fig. 3. Architecture of the demonstration platform

IV. Lessons Learned and Discussion

The convergence between Trusted Execution Environments and Opportunistic networks, that we named Edgelet computing, has not been deeply studied yet. First, this demonstration illustrates a potential usage of this new paradigm on top of a real medical use-case being deployed in the field. It demonstrates that large-scale general-purpose computations can be performed over a set of weakly connected devices while providing high security guarantees. Edgelet computing leverages the TEE security to perform computations on clear-text data (once decrypted locally). It can then combine computation generality and scalability contrary to homomorphic encryption, secure multiparty computation or local differential privacy solutions. Second, the demonstration shows critical parts of the Edgelet computing internals related to (i) privacy preservation, thanks to the support of horizontal and vertical partitioning, and (ii) resiliency, thanks to the Overcollection strategy, well adapted to sampling queries. While Overcollection applies if the processing is distributive, backup strategy can be used in all other situations, though at the price of a higher complexity and lower performance. Hence, we firmly believe that Edgelet computing opens up important opportunities in terms of personal data management.

References

[1] "Cisco Annual Internet Report (2018–2023) White Paper," Cisco, Mar. 2020. https://tinyurl.com/cisco-internet-report

[2] L. Javet, N. Anciaux, L. Bouganim, and P. Pucheral, "Edgelet Computing: Pushing Query Processing and Liability at the Extreme Edge of the Network," in CCGrid 2022, Taormina, Italy, 2022, pp. 160–169.

[3] D. Boneh, C. Gentry, S. Halevi, F. Wang, and D. J. Wu, "Private Database Queries Using Somewhat Homomorphic Encryption," in ACNS, Banff, AB, Canada, 2013. Proceedings, 2013, vol. 7954, pp. 102–118.

[4] J. Bater, G. Elliott, C. Eggen, S. Goel, A. N. Kho, and J. Rogers, "SMCQL: Secure Query Processing for Private Data Networks," Proc. VLDB Endow., vol. 10, no. 6, pp. 673–684, 2017.

[5] C. Dwork, "Differential Privacy," in Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II, 2006, vol. 4052, pp. 1–12.

[6] "DomYcile Project: https://tinyurl.com/domycile, Salon E-Tonomy: https://tinyurl.com/e-tonomy."

[7] M. Sabt, M. Achemlal, and A. Bouabdallah, "Trusted Execution Environment: What It is, and What It is Not," in 2015 IEEE Trustcom/BigDataSE/ISPA, Aug. 2015, vol. 1, pp. 57–64

[8] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks," IEEE Commun. Mag., vol. 44, no. 11, pp. 134–141, 2006.

[9] F. Tramer, F. Zhang, H. Lin, J.-P. Hubaux, A. Juels, and E. Shi, "Sealed-Glass Proofs: Using Transparent Enclaves to Prove and Sell Knowledge," in EuroS&P, 2017, pp. 19–34.

[10] D. Sculley, "Web-scale k-means clustering," in Proceedings of the 19th international conference on World wide web - WWW '10, 2010, p. 1177.

[11] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," in SIMUTools '09, New York, NY, USA, 2009.

[12] "Group By Grouping Sets, Snowflake." https://tinyurl.com/grouping-sets