Leveraging Secure Multiparty Computation in the Internet of Things

Marcel von Maltitz and Georg Carle
Technical University of Munich
Boltzmannstraße 3
Garching b. München
{lastname}@net.in.tum.de

ABSTRACT

Centralized systems in the Internet of Things—be it local middleware or cloud-based services—fail to fundamentally address privacy of the collected data. We propose an architecture featuring secure multiparty computation at its core in order to realize data processing systems which already incorporate support for privacy protection in the architecture.

1. INTRODUCTION

Smart environments and smart buildings constitute a vital part of the Internet of Things. In these contexts, sensors are deployed to gather information about the state of the real-world environment. This information, in turn, represents the data foundation for services that influence the environment state, provide insights for inhabitants and interact with them. Examples for these services are public displays, which give statistical information about the building state, monitoring services for maintenance personnel and anomaly detection systems which detect incidents and failures.

These and many other services have in common that they do not directly work on the raw data gathered by the sensors. Instead, they use derived aggregated results by computational preprocessing: Public displays show diagrams of statistical data, monitoring and anomaly detection services work with events and alerts gained by rules, machine learning or other types of computation.

For mediating the data flow between the sensor platforms—the data sources—and the services—the data consumers—typically a middleware is deployed. Its purpose encompasses collection and storage of raw data, analysis, processing and finally forwarding the obtained results to the data consuming services. This middleware can either be a local part of the smart environment but can also be provided as cloud service.

This type of architecture and the corresponding handling of data has severe implications for the privacy of the sensor data: **1)** The middleware acts as a third party which gains full access to raw data coming from the sensors. This third party might not even be under control of the administrators of the smart environment and hence untrustworthy. **2)** By pushing data to a third party, sources lose insights into how their data is used afterwards. Data processing becomes intransparent for them. **3)** Similarly, sources lose control over the usage of their data. Especially, revocation of data requires trust in the data holder to actually obey. **4)** Even if trustworthy, the third party is still a high value target for attackers.

2. PRIVACY PRESERVING DATA PROCESSING

Our vision is to realize the described functionality while fundamentally providing privacy protection on the architectural level. We propose that raw data created by the distributed sources is not collected by a middleware but remains distributed on these sources. This allows secure computations and can make consent and cooperation of the sources a necessity for the execution.

Our understanding of privacy and data protection is based on [1, 2]. They most importantly feature the protection goals of *data minimization*, *unlinkability*, *transparency*, and *intervenability*. Against this background, the positive implications of our approach are as follows: The amount of data in the system is minimized since there are no intermediaries which can also access data. Logically, the derived results are directly transmitted from the sources to the final consumers. The potential for data misuse and unauthorized recombination of data is decreased since data of different sources is not stored at the same logical place in a linkable fashion. Specifically, only making allowed computations technically possible concomitantly realizes purpose binding. The required cooperation of the data sources in turn provides them with information about the ongoing computations and the usage of their data. This constitutes transparency, especially when this feedback is enhanced with meta information about the final consumer. Persisting these insights can additionally realize accountability. Lastly, given the cooperation requirement and aforementioned transparency, they remain in control since they can specifically decide beforehand whether to cooperate and

to provide their data for the usage in question or not.

3. ARCHITECTURE DESIGN

The provided vision satisfies several data protection goals which are not yet fulfilled by state-of-the-art architectures. In order to realize this vision technically, the following main challenge has to be addressed: It must be possible to derive computation results from raw data of different sources without sharing this data among them nor handing it to a third party for computation.

For this purpose, secure multiparty computation (SMC) [3–5] can be employed. Instead of local computations of a third party a secure protocol among the sources is executed [6, 7]. Afterwards each only knows its own input and the final output of the computation. All exchanged intermediary data due to technical reasons does not allow recovering other parties inputs. Mathematical foundations for realizing arbitrary functions as SMC invocations are known since the 80's [8, 9] but protocol improvements for security and performance [10, 11] and new applications [12] are still current research.

For sucessfully applying SMC in smart environment we propose the following architecture: The formerly stated middleware is replaced by a *gateway*. The vital difference is that the gateway does not obtain access to the raw data of the sources. Instead, facing the sources it only fulfills management and orchestration purposes to carry out SMC computations. Towards the consumers, it presents an API which abstracts from SMC and resembles an interface a centralized middleware would provide.

Robust automated execution of SMC [13].

The main purpose of the gateway is to handle SMC sessions in cooperation with the sources. For this, the gateway must be initially known by them. Similarly, upon connection interruption or due to churn of mobile sources a present gateway has to be redetected. This is realized by a service discovery technology like mDNS [14, 15]. After detection, a setup between the new source and the gateway is performed: The gateway is informed about data and computation protocols provided by the new source. This data constitutes a state about currently obtainable insights about the environment in the form of a metadata directory. Furthermore, the gateway establishes a control channel to the source allowing to prepare and orchestrate SMC sessions.

The gateway specifies all aspects of an upcoming session and communicates them to the participating sources: The identity and the connection endpoints of cooperators, the data to be used for computation and the protocol to be executed. The computation itself is monitored by the gateway. On success, the gateway receives the result. If the computation fails, the gateway tries to recover or to fully restart the session. This is hidden from any consumer in any possible cases to achieve service character.

Data Requests and Access Control.

The purpose of the gateway towards the consumers is to mimic a standard middleware providing data upon request. Here, the metadata directory provides information to the consumers what data is obtainable at this point in time. This metadata should also abstract from SMC specifics allowing to post requests which already declare the aggregation result, e. g. "the average amount of individuals in floor 3.A of the building per hour". Receiving these requests, the gateway then transforms them into a corresponding SMC session and replies with the result afterwards.

Correct representation of data requests supports access control, transparency and intervenability essentially. We assume requests to be authenticated and integrity protected. The gateway is then able to perform access control and plausbility checks when examining the purpose of the request, the identity of the consumer and the type of requested data. During SMC session setup the gateway also transmits the original request of the consumer to each collaborator, consequently realizing request transparency for sources. Additional persisting the requests provides distributed request accountability. Lastly, this information can be evaluated by the sources before executing the computation. Each source can decide individually whether to contribute to the requested computation or not. In case a single source veotes against the computation, it cannot be executed; this is handled as a special, expected error by the gateway and can be addressed accordingly by it. In summary, we deliberately leverage the necessity of cooperation when performing computations to support the mentioned further privacy properties.

4. CONCLUSION

We presented a vision of privacy preserving data processing in dynamic environments. Our design features a management and orchestration middleware for secure multiparty computation which allows application of SMC as an adaptive and robust service. Furthermore, we show how the features of SMC can be complemented in order to fulfill further established privacy protection goals.

We see that fundamental innovation in system architecture allows more straightforward addressing of privacy goals. While also raising new challenges to be solved, they provide an alternative approach to establishing privacy as an afterthought in a predetermined system.

5. ACKNOWLEDGEMENTS

This work has been supported by the German Federal Ministry of Education and Research, project DecADe, grant 16KIS0538 and the German-French Academy for the Industry of the Future.

6. REFERENCES

[1] M. Rost and A. Pfitzmann, "Datenschutz-Schutzziele – revisited," *Datenschutz und Datensicherheit DuD*, vol. 33, no. 6, pp. 353–358, 2009.

[2] M. Hansen, M. Jensen, and M. Rost, "Protection Goals for Privacy Engineering," in *2015 IEEE Security and Privacy Workshops*, 2015, pp. 159–166.

[3] A. C. Yao, "Protocols for secure computations," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*. Washington, DC, USA: IEEE, 1982, pp. 1–5.

[4] ——, "How to generate and exchange secrets," in *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 1986, pp. 162–167.

[5] R. Cramer, I. B. Damgard, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*. New York, NY, USA: Cambridge University Press, 2015.

[6] R. Canetti, "Security and Composition of Multi-party Cryptographic Protocols," 1999.

[7] ——, "Universally composable security: a new paradigm for cryptographic protocols," *IEEE International Conference on Cluster Computing, SFCS*, pp. 136–145, 2001.

[8] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness Theorems for Non-Cryptographic Fault Tolerant Distributed Computation," *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing (STOC)*, pp. 1–10, 1988.

[9] D. Chaum, C. Crépeau, and I. Damgård, "Multiparty Unconditionally Secure Protocols," *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pp. 11–19, 1988.

[10] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Lecture Notes in Computer Science*, vol. 7417, 2012, pp. 643–662.

[11] M. Keller, E. Orsini, and P. Scholl, "MASCOT," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 830–842.

[12] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical Secure Aggregation for Privacy Preserving Machine Learning." *IACR Cryptology ePrint Archive*, vol. 2017, p. 281, 2017.

[13] M. von Maltitz, S. Smarzly, H. Kinkelin, and G. Carle, "A Management Framework for Secure Multiparty Computation in Dynamic Environments," in *NOMS 2018 – IEEE/IFIP DOMINOS Workshop*, Taipei, Taiwan, 2018.

[14] S. Cheshire and M. Krochmal, "Multicast DNS," RFC 6762 (Proposed Standard), Internet Engineering Task Force, Feb. 2013.

[15] ——, "DNS-Based Service Discovery," RFC 6763 (Proposed Standard), Internet Engineering Task Force, Feb. 2013.