Tolla: A user-isolating data management system

Aakash Sharma
UiT – The Arctic University of Norway
aakash.sharma@uit.no

ABSTRACT

Privacy of our personal data stored in clouds is under constant threat. Data from our activities online and even offline are being stored, processed and mined to generate valuable information for commercial interests. While this is being done, the individuals have very little or no knowledge about the sub-processors accessing their data. The lack of strict regulations has allowed commercial interests to disregard the privacy of users. A growing distrust among users about storing their personal data in the cloud has created the need for privacy preserving systems. In this direction, we present Tolla, a data management system for personal data. Tolla gives user total control over their data along with granular privacy controls to enable legacy internet services.

KEYWORDS

cloud, edge computing, security, privacy

1 INTRODUCTION

Internet services are constantly developing new features that capture and make use of personal data to understand our behavior, deliver relevant ads and what not. As our data is rapidly becoming a new commodity to be traded among large Internet companies, the importance of a stricter control is becoming more and more necessary and imminent. Recent studies [15, 16] have shown a growing distrust among individuals about their data stored at these third-parties.

The General Data Protection Regulation (GDPR)[1] of the European Parliament & Council is designed to enforce a stricter control over data processing and introduce more accountability. The regulations come into effect on 25^{th} May 2018. The regulations provide individuals with various rights over their own personal data, even if they are stored at cloud servers worldwide. The regulation enables individuals to be more aware of the processing done on their data and it will be possible only if they have consented to it.

2 PROBLEM DESCRIPTION

Most existing Internet services hide away most of their processing from the user. Their users are generally not aware of where their data is being stored and what processing is being done on it. The privacy controls provided to the end users typically fail to provide granular control over the data being processed. Popular protocols such as OAuth [12] enable cross-site data sharing via delegated API calls. However, the policies are defined by the web services using OAuth. If individuals created their own policies, strong assurances about how those policies are enforced are required [22]. The GDPR states that explicit consent is required for processing and users must be notified of any processing of their data. Another aspect of the GDPR that we are interested in, is the ability to switch between service providers while keeping user's data thus preventing data lock-in. This will enable a new generation of services which rely on a multitude of data collected by the user from various sources, while the user will remain in complete granular control of the data. One example of such a service can be seen in professional sports where fitness, nutrition, sleep and mood data can help athletes train better and even prevent suicides [20]. Some of this data can be considered sensitive in nature and require stricter controls depending upon the regional laws, individual and the privacy policies of the institution. We already provide services[2] in lifelogging sport domain for hundreds of elite athletes who are currently monitored, analyzed and intervened on a 24/7 basis. Another problem associated with an athlete's data is switching between different teams. The problem of ownership and reusing data collected previously arises due to data lock-in by the different systems in use.

3 RELATED WORK

One of the widely used methods for preserving privacy is anonymization. For example, in Prochlo [3], privacy is achieved by scrubbing all personal identifiable information from the browser statistics and mangling it into parts until no individual browser can be identified. The anonymization of health data in our case only makes sense in population-wide studies. For individual interventions, data must remain identifiable. The individual must be aware of the intention behind the processing of data and the services accessing the data. This is stipulated in the GDPR. It might be possible that the individuals do not want to store data in a cloud [16] at all. For privacy concerns, approaches like Databox [17] require data to be processed in an *edge node*.

Esposito et al. [6] argue that the use of edge computing eases data privacy challenges. However, the trade-off between performance, scalability, persistence, and reliability might not make it suitable for every application. While processing, storage and analysis of data can be leveraged using a cloud, mechanisms must be put in place to ensure privacy. On the other hand, edge computing provides privacy preserving processing, however it provides challenges such as cost efficiency, real-time scheduling etc. The work also outlines security challenges while dealing with edge computing as they can be relatively easily exploited by an adversary.

Sahi et al. [21] explored the state of research of privacy in e-Healthcare systems. Their work highlights the privacy concerns in healthcare data as it is extremely personal and private. And any

[1] eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32016R0679

[2] www.corporesano.no

12^{th} *EuroSys Doctoral Workshop, April 2018, Porto, Portugal*
2018.

mismanagement of such data can lead to complications for both the patient and the e-Healthcare enterprise. Their work also discusses how certain privacy requirements are given less importance by the service providers while designing systems. Even though strong legal regulations such as HIPAA [9] exist for medical data, these systems rely on Role Based Access Control (RBAC) [7] mechanisms to prevent any unauthorized access to the system. RBAC uses the identity of the user/application to limit or allow access to the data. The identity is associated with certain roles for which privacy policies are defined. For example, a health practitioner, a nurse, or a police personnel. In the context of a multitude of applications which exist today, the solution is not feasible as the role of every individual application or data processor is not clearly defined. For GDPR, each application or data processor has to define an intent which can be a complex set of permissions. The user may revoke some of these permissions rendering the use of role-based access not suitable for implementing a GDPR compliant system.

In Sieve [22], the use of Attribute Based Encryption (ABE) is limited to hiding the homomorphic keys and the metadata. Sieve relies on encrypted data scheme which is only exposed to applications with relevant/correct attributes as privacy policies. For any retraction or revocation of keys, re-encryption and re-labeling is required by design. Complex privacy policies such as access to location data only if it is more than one year old will be very difficult to implement in Sieve. Sieve assumes and provides access to every data element which has been consented by the user based on the label of the data. Granular control is not available in Sieve. In contrast, Tolla provides access to the data, only if the current state of privacy policy permits.

Similar to Sieve, approaches like [18, 19, 23] rely on homomorphic encryption [11] to provide privacy preserving computing in the cloud. Homomorphic encryption allows computation on encrypted data which prevents the *curious* cloud provider from learning anything about the data. However, it is also known to have some vulnerabilities. Security analysis such as [1, 4] demonstrate some known weakness which can lead to leaking of the keys. Moreover, it is well suited in cases where encrypted data is uploaded to the cloud for processing. However, as shown in recent studies [15, 16] there is a grown distrust about that. Sandboxing approaches like πBox [14] isolate a user's instance of an application from other instances and users. πBox defines certain communication channels which may suit some applications, however, some of the approaches require accuracy and privacy trade-offs. Moreover, the finite amount of communication channels assume that the applications and their permissions do not change over time. Applications are assigned a privacy rating according to their behavior and cannot evolve as the user's consent changes. There are no granular privacy controls defined in πBox. The GDPR is likely to bring more transparency on what our data is being used for. We believe that with transparency, the individuals will be more aware of privacy risks and will define their own privacy policies [5, 8]. Thus, there is a strong need for granular privacy controls.

4 OUR APPROACH

We have built Tolla, a data management system for personal data. Tolla runs in a container-based environment that is *location-agnostic*. It provides granular privacy controls similar to Databox [17] without the requirement of having an *edge node*. The privacy and access controls are guaranteed based on certificate authentication to the services which have been consented to by the individual. Extending further upon our previous work using *meta-code* [13], Tolla uses Ciphertext-Policy Attribute Based Encryption (CP-ABE) [2] as a mechanism to enforce any real-time changes in the consent or privacy policies. Our goal is to provide a framework that offers privacy and accountability by design. Application developers and data scientists can write their queries without having to worry about managing which users have consented or not.

The Tolla system is built to provide privacy guarantees on the data by isolation. The use-cases for Tolla will be from the lifelogging sport domain where we already provide services. Each individual's data is stored in a federated storage. This enables data to be moved with the individual. Our design does not dictate where data should be stored, locally, a distributed file system or a combination of both. However, we assume that it is possible to load data into the container in a secure way while bootstrapping. While bootstrapping, the data of an individual is loaded into a Personal Data Storage Unit (PDSU). Existing applications can utilize the Tolla framework without having to worry about managing *consent* from each individual. Existing data processing can be modified to point to Data Processing Units or DPUs for querying user data. Each DPU is capable of querying data stores while considering user's content. During our initial evaluation, we found out that a typical football team member's personal fitness and lifelogging data running inside Tolla can be processed on a laptop with decent amount of memory.

Every access to the data stored in a Personal Data Storage Unit (PDSU) is facilitated by the system and guarded by granular privacy controls. Any request for data by an application is routed through a Data Processing Unit (DPU). The DPU acts on behalf of the application and provides the *intent* behind accessing a particular set of data. The Certificate Authority (CA) verifies the identity of the application. There is no need for one global CA for every instance of Tolla. Similar to a certificate validation for SSL [10] in a browser, an individual's data storage unit can have more than one trusted CA. If the individual has consented to the data processing by the application for this *intent*, the CA signs the X.509 certificate of DPU. While signing the certificate, the CA fetches policies from the Policy Engine (PE) and adds them to the certificate along with the *intent* defined by the DPU. The CA also provides a decryption key based on Attribute Based Encyption to the DPU, which is used later. The Policy Engine(PE) also provides addressing mechanisms via CA to the DPU to access the available PDSU(s). After obtaining the signed certificate, a DPU can then query a PDSU for data. For opening a connection to a PDSU over TLS, the DPU provides the signed certificate. The *verifier* running inside a PDSU authenticates the received certificate from a DPU and creates a secure connection. The *verifier* then verifies the data being asked and whether it is compliant with the *intent* specified in the certificate. While the

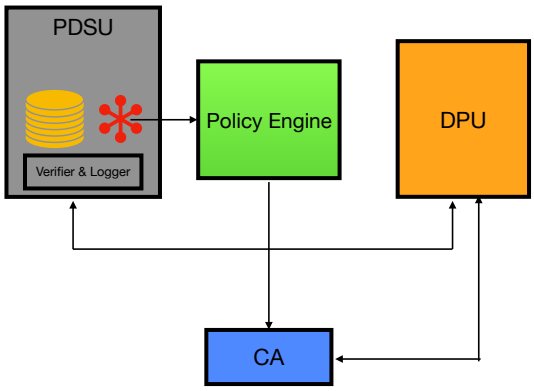

Figure 1: System Architecture

certificate is being verified, the *logger* logs the request along with key identifiers for audit purposes. After verification, the requested query is processed and the result is calculated. The calculated result is then encrypted using CP-ABE while using the set of parameters from the policy defined for the *intent* or *data fields* as the key. The encrypted result is sent back to DPU. The received results can be decrypted by DPU using the attributes from the signed certificate provided by the CA earlier.

4.1 Accessing the data

A Data Processing Unit (DPU) works on behalf of an application. Typical operations performed by a DPU are reading user data and performing analytics on it. When a Data Processing Unit (DPU) requests access to information based on an individual's data, there are two layers of authentication mechanisms in Tolla. The first mechanism comes in place which has been approved by the Certificate Authority (CA). The connection between a DPU and a Personal Data Storage Unit (PDSU) takes place over TLS. The DPU uses the signed certificate which it obtained from CA in order to open a connection to the PDSU. Only if the certificate matches the *intent* which the user initially consented to, the further steps of obtaining data can take place. In case the intent does not match or the certificate is not signed by a trusted CA, the connection is dropped and the event is logged. Once the DPU has been authenticated by the *verifier*, the DPU sends the required query over a secure connection. Upon receiving the query, the *logger* logs the query for auditing. If the received query matches the columns associated with the *intent*, the PDSU queries the database and obtains results. Before sending the results to the requesting DPU, an Attribute Based Encryption (ABE) scheme is used to encrypt the result. Depending upon the type of query and the sensitivity of the data and the DPU, the PDSU can employ a different set of attributes for the encryption. For example, aggregation queries without a time period filter can be considered less sensitive. In such case, the PDSU can use the 'application ID', 'certificate signing date' check as the set of attributes for encrypting the result. The use of Ciphertext-Policy ABE allows various logical evaluations which can be employed, such as 'certificate signing date' must be after 01-01-2018. Only if the DPU has valid key relating to the certificate it presented and a certificate which matches the signing date criteria, it will be able to see the results. For relatively more sensitive data, such as a stream of GPS coordinates from a fitness tracker or financial transactions for last month, the PDSU can employ more attributes in order to block decryption by unauthorized DPU. It is important that while deciding the attributes for CP-ABE, the PDSU picks relevant privacy policy stored in the PDSU for the requesting Data Processing Unit (DPU).

The signed certificate requirement by *verifier* ensures that only verified third-party services will be able to access an individual's data. Only when the individual has explicitly *consented* to the *intent* of a third-party, will the request be processed. The use of CP-ABE ensures that the real-time enforcement of consent and privacy policies. If the policy has been modified or consent has been revoked, it will be first reflected in the privacy policies stored in the PDSU. The PDSU then pushes policies to the Policy Engine (PE) so that those can be reflected in the certificates signed by CA later. However, if a request is under processing and the individual decides to revoke consent, the encryption scheme ensures that the DPU and subsequently the application fails to obtain results. This is a result of encrypting scheme (CP-ABE) utilizing the attributes from the consent and policy available at the time of processing the result of the query. The decryption fails as the available attributes in the decryption key with the DPU fail to match the ones used in the encryption. In case of such failure, the DPU can approach the Certificate Authority (CA) again and obtain a new set of signed certificate and decryption key with the updated privacy policy. If the policy remained unchanged, the DPU is able to decrypt the result using the key it obtained earlier while obtaining a signed certificate from the CA. The verifier is also responsible for translating *intent* to database fields and support for aggregation bounds defined in the policies in a PDSU.

5 CONCLUSION

We have introduced Tolla, a location-agnostic framework for personal data management. Tolla demonstrates compliance for a set of GDPR requirements. Tolla allows the same level of privacy as many edge-based systems. Access control and logging for accountability can be done on a per-user granularity without having to replicate complex filtering logic in data processing units that read data from a personal data storage unit (PDSU). The storage abstractions are of significance in professional sports leagues where athletes can now control access to their data. The athletes can also move their data between teams as they do.

REFERENCES

[1] Mikhail Babenko, Nikolay Chervyakov, Andrei Tchernykh, Nikolay Kucherov, Maxim Deryabin, Gleb Radchenko, Philippe OA Navaux, and Viktor Svyatkin. 2018. Security analysis of homomorphic encryption scheme for cloud computing: Known-plaintext attack. In *Young Researchers in Electrical and Electronic Engineering (EIConRus), 2018 IEEE Conference of Russian*. IEEE, 270–274.

[2] John Bethencourt, Amit Sahai, and Brent Waters. 2007. Ciphertext-policy attribute-based encryption. In *Security and Privacy, 2007. SP'07. IEEE Symposium on*. IEEE, 321–334.

[3] Andrea Bittau, Ulfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. 2017. PROCHLO: Strong Privacy for Analytics in the Crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles*. ACM, 441–459.

[4] Sonia Bogos, John Gaspoz, and Serge Vaudenay. 2018. Cryptanalysis of a homomorphic encryption scheme. *Cryptography and Communications* 10, 1 (2018), 27–39.

[5] George Danezis, Josep Domingo-Ferrer, Marit Hansen, Jaap-Henk Hoepman, Daniel Le Metayer, Rodica Tirtea, and Stefan Schiffner. 2015. Privacy and Data Protection by Design-from policy to engineering. *arXiv preprint arXiv:1501.03726* (2015).

[6] Christian Esposito, Aniello Castiglione, Florin Pop, and Kim-Kwang Raymond Choo. 2017. Challenges of connecting edge and cloud computing: A security and forensic perspective. *IEEE Cloud Computing* 4, 2 (2017), 13–17.

[7] David Ferraiolo, Janet Cugini, and D Richard Kuhn. 1995. Role-based access control (RBAC): Features and motivations. In *Proceedings of 11th annual computer security application conference*. 241–48.

[8] Simone Fischer-Hübner, Julio Angulo, and Tobias Pulls. 2013. How can cloud users be supported in deciding on, tracking and controlling how their data are used?. In *IFIP PrimeLife International Summer School on Privacy and Identity Management for Life*. Springer, 77–92.

[9] Centers for Disease Control, Prevention, et al. 2003. HIPAA privacy rule and public health. Guidance from CDC and the US Department of Health and Human Services. *MMWR: Morbidity and mortality weekly report* 52, Suppl. 1 (2003), 1–17.

[10] Alan Freier, Philip Karlton, and Paul Kocher. 2011. The secure sockets layer (SSL) protocol version 3.0. (2011).

[11] Craig Gentry. 2009. *A fully homomorphic encryption scheme*. Stanford University.

[12] Dick Hardt. 2012. The OAuth 2.0 authorization framework. (2012).

[13] Håvard D Johansen, Eleanor Birrell, Robbert Van Renesse, Fred B Schneider, Magnus Stenhaug, and Dag Johansen. 2015. Enforcing privacy policies with meta-code. In *Proceedings of the 6th Asia-Pacific Workshop on Systems*. ACM, 16.

[14] Sangmin Lee, Edmund L Wong, Deepak Goel, Mike Dahlin, and Vitaly Shmatikov. 2013. πBox: A Platform for Privacy-Preserving Apps.. In *NSDI*. 501–514.

[15] Chantal Lidynia, Philipp Brauner, and Martina Ziefle. 2017. A Step in the Right Direction–Understanding Privacy Concerns and Perceived Sensitivity of Fitness Trackers. In *International Conference on Applied Human Factors and Ergonomics*. Springer, 42–53.

[16] Byron Lowens, Vivian Genaro Motti, and Kelly Caine. 2017. Wearable Privacy: Skeletons in The Data Closet. In *Healthcare Informatics (ICHI), 2017 IEEE International Conference on*. IEEE, 295–304.

[17] Richard Mortier, Jianxin Zhao, Jon Crowcroft, Liang Wang, Qi Li, Hamed Haddadi, Yousef Amar, Andy Crabtree, James Colley, Tom Lodge, et al. 2016. Personal Data Management with the Databox: What's Inside the Box?. In *Proceedings of the 2016 ACM Workshop on Cloud-Assisted Networking*. ACM, 49–54.

[18] Zhan Qin, Jian Weng, Yong Cui, and Kui Ren. 2018. Privacy-Preserving Image Processing in the Cloud. *IEEE Cloud Computing* (2018).

[19] Amitesh Singh Rajput and Balasubramanian Raman. 2018. CryptoCT: towards privacy preserving color transfer and storage over cloud. *Multimedia Tools and Applications* (2018), 1–23.

[20] Ashwin L Rao and Eugene S Hong. 2015. Understanding depression and suicide in college athletes: emerging concepts and future directions. (2015).

[21] Muneeb Ahmed Sahi, Haider Abbas, Kashif Saleem, Xiaodong Yang, Abdelouahid Derhab, Mehmet A Orgun, Waseem Iqbal, Imran Rashid, and Asif Yaseen. 2018. Privacy Preservation in e-Healthcare Environments: State of the Art and Future Directions. *IEEE Access* 6 (2018), 464–478.

[22] Frank Wang, James Mickens, Nickolai Zeldovich, and Vinod Vaikuntanathan. 2016. Sieve: Cryptographically Enforced Access Control for User Data in Untrusted Clouds.. In *NSDI*, Vol. 16. 611–626.

[23] Zhihua Xia, Xinhui Wang, Liangao Zhang, Zhan Qin, Xingming Sun, and Kui Ren. 2016. A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing. *IEEE Transactions on Information Forensics and Security* 11, 11 (2016), 2594–2608.