## Privacy Engineering at Scale

Amongst the ever-evolving landscape of privacy, the only constant is an agreement on building privacy in by design. The implementation of this axiom requires organizations to adopt, build, and deploy a new set of privacy engineering practices alongside their product development lifecycle. Ideally these practices are prioritized towards the left of this lifecycle to build privacy in by default. Therefore, these practices and associated privacy enhancing technologies need to be mature, broadly applicable, and easy to operationalize.

Yet many of the privacy enhancing technologies that gain traction in the privacy engineering research community do not fulfill these criteria. Consider differential privacy, synthetic data generation, or homomorphic encryption. Neither is broadly applicable as they solve distinct privacy problems for specific product use cases. For example, it is hard to construct a one size fit all synthetic data solution that can be used by developers, as different data types may require a distinct model for synthetic data generation due to the underlying differences in the properties of the data. Similarly, homomorphic encryption has not yet matured enough to provide the comparable computational speeds as operations over plaintext data and furthers the challenges of key management. Finally, the limits of differential privacy were recently documented in the US government's decision to pull back from using the technology for American Community Survey.

Yet there are technologies that can be used to build privacy through engineering that are less discussed. For example, consider privacy threat modeling. Threat modeling is an excellent way to build privacy in as it looks to identify privacy threats at an architecture level, before any code is written. Kim Wuyt and other researchers from University of Leuven have designed a LINDDUN a privacy threat modeling framework. This approach has been further extended by Jayati et al. into MAP or Models of Applied Privacy, which integrates a persona-based approach to threat modeling reducing the need for privacy expertise and leveraging a solutions that developers and product managers are already familiar with. Kristen et al. have further suggested the use of rules-based solutions like Threagile to automate aspects privacy threat modeling.

Another example is static code-analyses for privacy. Once threat modeling is done and developers begin to write code, static code analyses is often used in security to identify security vulnerabilities. A similar approach can be used for privacy. Vendors like Privado.ai are offering open-source solutions for code written in Java. However, others can potentially build these capabilities from scratch using solutions like CodeElf and Graph4Code. This allows product teams to identify what data is being collected, where it is being collected, and where it is sent – for each and every pull request. This visibility allows for better privacy impact assessments.

Finally, once a product is ready for shipping it may be possible to do privacy assessments, similar to security assessments, using dynamic code analysis for privacy. For example, researchers have detected privacy violations like COPPA violations by analyzing data flows at runtime in Android mobile applications. If used by developers and product teams a similar approach can be used to validate compliance against specific privacy requirements.

Much of the discourse around privacy enhancing technologies is limited to solutions with limited applications and requiring further maturity. This paper, instead, highlights the emerging set of developer-oriented privacy enabling technologies. First, we discuss applied privacy threat modeling and scaling it using rules-based engines. Second, we discuss static code analysis tools and how they can be leveraged for detecting privacy violations. Finally, we discuss dynamic code analysis tools and how they can be used to validate privacy requirements.

Bibliography


1. Domingo-Ferrer, J., Sánchez, D. and Blanco-Justicia, A., 2021. The limits of differential privacy (and its misuse in data release and machine learning). *Communications of the ACM*, *64*(7), pp.33-35.
2. Wuyts, K., Sion, L. and Joosen, W., 2020, September. Linddun go: A lightweight approach to privacy threat modeling. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* (pp. 302-309). IEEE.
3. Dev, J., Rashidi, B. and Garg, V., 2023. Models of Applied Privacy – A Persona Based Approach to Privacy Threat Modeling. In 2023 ACM Computer Human Interaction. ACM.
4. Tan, K. and Garg, V. 2022. Privacy Shift Left – A Machine Assisted Threat Modeling Approach. Usenix Privacy Engineering, Practice, and Respect.
5. https://www.privado.ai/open-source
6. https://unbug.github.io/codelf/
7. https://github.com/wala/graph4code
8. Reyes, I., Wijesekera, P., Razaghpanah, A., Reardon, J., Vallina-Rodriguez, N., Egelman, S. and Kreibich, C., 2017, May. " Is Our Children's Apps Learning?" Automatically Detecting COPPA Violations. In *Workshop on Technology and Consumer Protection (ConPro 2017), in conjunction with the 38th IEEE Symposium on Security and Privacy (IEEE S&P 2017)*.
9. Sanfilippo, M.R., Shvartzshnaider, Y., Reyes, I., Nissenbaum, H. and Egelman, S., 2020. Disaster privacy/privacy disaster. *Journal of the Association for Information Science and Technology*, *71*(9), pp.1002-1014.