2019 IEEE Security and Privacy Workshops (SPW)

A Privacy-Aware V-Model for Software Development

Ala'a Al-Momani*, Frank Kargl*, Robert Schmidt†, Antonio Kung‡, Christoph Bösch*
*Institute of Distributed Systems, Ulm University, Ulm, Germany
{alaa.al-momani, frank.kargl, christoph.boesch}@uni-ulm.de
†Denso Automotive Deutschland GmbH, Munich, Germany
r.schmidt@denso-auto.de
‡Trialog, Paris, France
antonio.kung@trialog.com

***Abstract*—After the adoption of new data protection regulations, like GDPR, proper treatment of privacy throughout the system development lifecycle has become a must. In this paper, we discuss several aspects to more easily and effectively integrate privacy engineering in system development and how to bring the notion of privacy-by-design into practice. We propose the new *W-model* as a privacy-aware extension of the V-model frequently used in software engineering. One stage of the W-model deals with analyzing privacy in the system where privacy engineers conduct a privacy impact assessment in order to elicit privacy threats and to find a suitable countermeasure to remedy each threat. With respect to finding suitable countermeasures, we provide requirements the countermeasures need to meet in order to be selected. In addition, we introduce a *cost* function that assists privacy engineers in selecting the most suitable countermeasure. Furthermore, we point out several open issues that future work needs to address.**

I. Introduction

The era of big data puts Service Providers (SPs) in a position to collect, process, store and share an enormous amount of users' data. Generally, collecting, processing, storing, and sharing data between several parties yield a better functionality of services, provide better decision making, and more accurate estimation of, e.g., future demand. As a downside of these benefits, users' privacy is put at risk. In particular, SPs frequently collect sensitive data and Personally Identifiable Information (PII), from which further sensitive information may be inferred about data subjects. To cope with this situation, the development of new laws and regulations to control the related operations on personal data (including collecting, processing, storing, and sharing) became necessary. The recent European General Data Protection Regulation (GDPR), which is enforced since May 2018, has made a world-wide impact way beyond Europe. In Japan, the amended Japanese Act on the Protection of Personal Information (APPI) is in full effect since May 2017. Such regulations are essential to define how and for what purpose SPs are entitled to collect, process, and share individuals information.

One of the aspects the GDPR strongly highlights is data-protection-by-design. Data-protection-by-design implicitly encourages both academia and industry to advance the research in the field of *Privacy-by-Design (PbD)* as PbD is seen as data-protection-through-technology design [1]. PbD requires a sound privacy engineering methodology to be followed throughout the development phases of any system or application. The benefit of such a methodology summarizes in translating and mapping the high-level requirements acquired from a specific regulation, e.g., GDPR, onto a more detailed level consisting of technical requirements and measures.

Engineering PbD has been addressed in the literature [2–5]. However, the applicability of such an engineering process is still questionable [6–8]. To enhance the applicability of a privacy engineering methodology, it is often integrated into software development lifecycle (SDLC) forming a holistic engineering methodology to introduce privacy-enhanced systems. Privacy, in general, and PbD have been discussed to be integrated into common SDLCs including the Agile [4, 9] and the Waterfall models [4].

One of the old but still very common SDLC models that is widely used among, e.g., the automotive industry, is the V-model [10–12]. The V-model is preferred to be used in scenarios where clear phases of the development process are required with clear documentation of each phase for, e.g., safety purposes. As the V-model has not been discussed with the consideration of privacy engineering and PbD, we introduce a privacy-aware V-model, which we call the *W-model*. The proposed W-model provides both system and privacy engineers following the V-model for software development with a systematic approach to integrate the PbD into the development process in order to introduce privacy-enhanced systems.

An essential part of the W-model is conducting a Privacy Impact Assessment (PIA). The goal of conducting such a process is two-fold; 1.) to elicit privacy threats, and 2.) to find the corresponding technical measures to mitigate the elicited threats. One of the common and widely used PIA processes is LINDDUN [13]. LINDDUN elicits privacy threats in a process called the problem-space and suggests the corresponding, potential, mitigation measures in a process called the solution-space. The problem-space has been further improved [14–18] by, mainly, enriching the process of threat elicitation with knowledge on the system level including the interactions among system components. This consideration results in improving the accuracy of the elicited threats [14].

DOI 10.1109/SPW.2019.00028

Enhancing the solution-space process has not been tackled yet even though such a process is a key to introduce a resilient privacy-enhanced design of the system. In order to enhance the process of the solution space, we provide critical discussion including formal requirements a countermeasure (CM) needs to meet in order to be selected. Such requirements help in the transition to the process of designing a privacy-enhanced architecture (PEAR) [19].

The rest of this paper is organized as follows: we provide a brief discussion on both the V-model and PRIPARE in Section II, which serves as the basis of the proposed W-model in Section III. Section IV critically discusses the privacy-centric stages within the W-model. Furthermore, we point out the open issues future work should consider in Section V before concluding the paper in Section VI.

II. Background

We now briefly describe the bases of the W-model; the V-model as a SDLC, and PRIPARE as a PbD methodology.

A. The Classical V-Model

The V-model [20] is a generic software development model that addresses several steps of the development process. It is widely used, e.g., in the automotive industry [10–12], because of its simplicity and clear documentation phases which make tracking issues during the development process easier. Generally, the steps of system development based on the V-model are divided into two main categories; the *system definition* steps and the *verification* steps. In system definition, system developers define the system requirements that properly reflect the high-level business requirements. Thereafter, system engineers come up with both a high- and low-level design of the system. Then, the implementation phase takes place before the verification steps start. Verification steps include testing at both unit and system levels. The verification steps are to ensure that the functionality the implemented system offers meets the requirements defined at earlier stages of the development. If any of the requirement is not met, then system developers are required to modify the design to ensure that all requirements are met. Upon accepting and releasing the system, system's operations are examined and maintained throughout its lifecycle.

B. PRIPARE

We take PRIPARE [4] as an example of a well-established PbD methodology with clear phases. Our goal is to address PbD phases and reflect them in the V-model so that developing privacy-preserving systems based on the V-model becomes easier and more systematic. PRIPARE recognizes 8 main phases of the PbD methodology; *analysis, design, implementation, verification, release, maintenance, decommissioning, and environment & infrastructure*.

The *analysis* phase includes addressing high-level requirements and discovering privacy issues through conducting a PIA. The *design* phase refers to defining an architecture that meets the predefined (functional and non-functional) requirements. The *implementation* phase refers to implementing the pre-designed PEAR. Moreover, the implemented system is verified against the requirements during the *verification* phase of PRIPARE. Furthermore, PRIPARE's phase of *release* points out the necessity of a response plan upon releasing the system. *Maintenance and decommissioning* deals with periodically assessing the system after the release, and implementing the necessary CMs in case of any new requirement. Finally, PRIPARE's phase of *environment & infrastructure* addresses advocating and enhancing the privacy-awareness among the organization's units.

III. The W-Model

Privacy and privacy-related activities are not specifically addressed in the classical V-model. Moreover, the V-model has not been addressed in PbD methodologies like PRIPARE [4]. In this section, we present our proposal for a privacy-aware variant of the V-model: the *W-model*. The W-model integrates PbD into the original V-model and thus suggests modifying the original stages while also introducing new stages to the V-model that explicitly address privacy aspects. Our proposed W-model is shown in Figure 1.

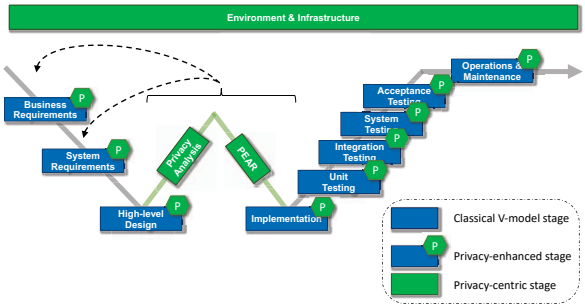

Fig. 1: The W-Model

The W-model suggests to fundamentally change the classical stages of the V-model to consider privacy. This reflects the *environment & infrastructure* phase PRIPARE proposed which promotes privacy awareness among the organization and throughout the whole lifecycle of a system. Considering privacy in such a way fosters the understanding of PbD within an organization, and enhances the efficiency of the development of privacy-preserving systems. This is due to considering privacy already at the early stages of development, i.e., at business requirements and system requirements. Besides considering privacy during each phase of the system lifecycle, we foresee a necessity of introducing impartial stages that explicitly address privacy in the system lifecycle. These stages are the *privacy analysis* stage which includes the PIA, and the *PEAR* which includes both privacy-preserving high-level design, and privacy-preserving low-level design. In the following, we briefly describe each stage of the W-model.

- *Business Requirements*: This stage represents forming the business requirements out of an idea developed by the

management or by public demand. The W-model suggests to, informally, consider user's privacy at this early stage in such a way to change obvious privacy-violating requirements to more privacy-considered requirements. With the help of the privacy-aware environment established within the organization, this stage might explicitly include privacy requirements as part of the business requirements, e.g., to comply with a specific regulation such as GDPR.

- *System Requirements*: System engineers keep user's privacy in mind while mirroring the business requirements into the system requirements and report any potential violations of privacy to the higher business units. Moreover, they document their findings of potential privacy violations to the subsequent stages' teams so that these violations are taken care of. System engineers are not required to perform any formal PIA at this stage. Conducting a PIA is one of the later tasks of privacy engineers that needs to be addressed at the *privacy analysis* stage. Privacy engineers are system engineers whose explicit task is addressing privacy in the system development.
- *High-level design*: System engineers construct a high-level design of the system requirements. This design is a *somewhat privacy-preserving design* of the system. In particular, this somewhat privacy-preserving design serves as an initial design to enable privacy engineers performing privacy analysis and further refining the design to be fully privacy-preserving in the subsequent stages.
- ***Privacy Analysis***: This stage is the first newly introduced stage in the W-model which is, entirely, privacy-centric. At this stage, a team (or teams depending on the scale of the system) consisting of privacy engineers and legal advisors, analyze the initial design of the system so that it is privacy-preserving and regulation-compliant. Analyzing privacy in the system summarizes in conducting a PIA for which an initial design of the system is required. This is where the previous *somewhat privacy-preserving design* of the system serves as the initial design that needs to be further analyzed at this stage. In particular, the task of the privacy engineers, at this stage, is two-fold: 1.) to elicit privacy threats in the initial design of the system, and 2.) to find suitable CMs for the elicited threats. Furthermore, throughout this step, privacy engineers conclude privacy requirements which need to be reflected in the initial design of the system through the selected CMs. Often, the defined privacy requirements at this stage conflict with either business or system requirements. In such a case, privacy engineers report the conflicting requirements to the corresponding team to find appropriate solutions to such conflicts.
- ***Privacy-Enhanced Architecture (PEAR)***: The stage of designing *PEARs* [19] is divided into two sub-stages:
  - *Privacy-Preserving High-Level Design*: At this sub-stage, a team of privacy and system engineers together address both the defined system and privacy requirements and reflect them in the previous — somewhat privacy-preserving — design. The focus here is restricted to the high-level components of the system and the interaction among them.
  - *Privacy-Preserving Low-Level Design*: Here, both system and privacy engineers construct a lower abstraction of the privacy-preserving high-level design of the system. That is, addressing and designing the low-level components of each of the high-level ones. Therefore, the design at this point is suitable to start the implementation process in the next stage.

  The goal of these two design phases is to come up with a privacy-enhanced architecture [19] that meets both pre-defined business and privacy requirements (or the balanced requirements in case of a conflict).
- *Implementation*: At this stage, system developers implement the designed PEAR of the system. An interaction between system developers and privacy engineers may take place to clarify some critical aspects with respect to privacy. Thus, prior knowledge of privacy through the enhanced privacy culture within the organization — as addressed by the *environment & infrastructure* — reduces the, sometimes, unnecessary interaction between developers and privacy engineers. This interaction has the potential to decrease the time-efficiency of the overall development process. Therefore, limiting this interaction to include only the necessary aspects, enhances the time-efficiency of the overall development process.
- *Unit Testing*: The goal here is to make sure that the low-level components execute as intended including the designed privacy-related components.
- *Integration Testing*: Here, a team of system verifiers (including both system and privacy engineers) ensures that the interaction of the low-level components executes as intended without posing privacy risks in the system.
- *System Testing*: System verifiers, at this point, ensure that the implemented system meets both the predefined system and privacy requirements while reporting any mismatch to the corresponding team.
- *Acceptance Testing*: This is to ensure that the system, which is about to be released to the public, meets the business requirements (including privacy and compliance requirements) defined at an earlier stage of development. Upon accepting the designed system, the developing organization releases the system while establishing comprehensive practices as a response in case of incidents such as, e.g., data breaches.
- *Operations & Maintenance*: This stage deals with maintaining the operation of the system while capturing newly emerging requirements including privacy-related ones through a periodic legal and technical assessment. Hence, their implication is incorporated in the system. Furthermore, in case of a privacy incident such as, e.g., a data breach, the response plan created upon releasing the system has to be conducted accordingly.

We conclude here that the W-model incorporates *all* PbD phases PRIPARE suggested, and integrates them into the V-model SDLC. We foresee the W-model to be used in organizations that follow the V-model for software development and that wish to design and introduce privacy-preserving systems.

IV. Analysis of The Privacy-Centric Stages

We restrict our focus to be on the privacy-centric stages, shown in Figure 2, and we point out some open issues that need to be investigated. We further provide first insights on possible approaches toward a systematic solution that potentially leads to a robust privacy-enhanced system design.

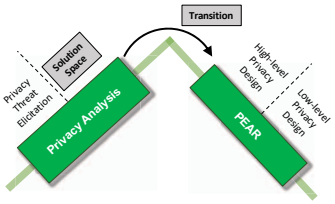

Fig. 2: Our focus in the rest of this paper is 1.) the solution space and 2.) the transition to PEARs.

A. Critical Issues of The Privacy-Centric Stages

The issues we report in this paper summarize mainly in the handover from the defined privacy threats in the privacy analysis to the PEAR. This handover includes the sub-stage of the solution-space and the transition to the PEAR. In particular, we define the following issues:

- In the solution space: *Selecting one CM to each elicited threat*. Most PIA methodologies, including LINDDUN [13], suggest a set of possible solutions — CMs often in the form of Privacy Enhancing Technologies (PETs) — to address one privacy threat. The issue, therefore, is that privacy engineers struggle in selecting **one** CM to **one** elicited threat. Furthermore, the struggle increases in case of a large-scale system in which many privacy threats are elicited and, thus, several sets of possible CMs are present.
- In the transition to PEARs: Often, the only aspect that is considered in selecting a CM (out of the possible set of CMs) is its fit to the elicited threat. The system level, as well as the scenario the threat applies in, are often disregarded in the CM-selection process. This may lead to the issue that the selected CM in the solution space process is unsuitable to be used in the system, despite the fact that, the same CM remedies the effect of a similar threat in other scenarios. For example, secure multiparty computation remedies the threat of information disclosure at a process, however, it is not applicable in scenarios where only one entity is foreseen in the system.

Those issues have not been addressed nor pointed out by previous work as the focus was mainly on enhancing the threat elicitation and risk assessment processes [15–18]. We see the solution-space and the transition to a PEAR as keys to come up with a sound privacy-preserving system. Well-structured processes in these aspects bring privacy engineering and PbD into an easier deployment.

Based on the previously defined issues in both the solution-space and in the transition to PEARs, we draw the following requirements for the CMs to be selected:

- *Correctness*. The selected CM remedies the risk of the elicited privacy threat.
- *Integrability*. The selected CM is integrable in the system environment. An example of a non-integrable CM includes a computationally-expensive fully homomorphic encryption scheme that is foreseen to be used in time-critical applications, e.g., real-time monitoring and processing of health data.
- *Compatibility*. The selected CM is compatible with other system components including CMs selected for other privacy threats. An example of incompatible CMs includes *encrypting a database* (in order to solve a threat of identifiability at that database) whereas the content is required in a different format to execute a *multiparty computation* scheme (to solve a threat of identifiability at a process within that system).

The intuition of considering these requirements is enriching the processes of the solution-space and the transition to PEARs with additional information on the system alongside its components' interaction. Particularly, the previous requirements take into consideration: 1.) the functionality and the goal the system aims to achieve, and 2.) the context of the system including both the scenario where the system is going to be applied, and the other technologies used at other parts of the system.

B. Toward Privacy-Enhanced Architectures

Despite that the previous requirements (*correctness, integrability*, and *compatibility*) help in excluding unsuitable CMs, there is a potential that privacy engineers still face several CMs, each meets all the requirements of *correctness, integrability*, and *compatibility*. Thus, it is still ambiguous — and considered an open issue — how to eventually select *one* CM to remedy *one* privacy threat.

As this may pose significant confusion among practitioners [21], we provide initial insights toward a more consistent approach to solving this particular issue. For this purpose, we propose the methodology shown in Algorithm 1. The proposed algorithm takes as input the *set* of possible CMs and outputs the most suitable CM. At first, privacy engineers sort the available CMs ascendingly according to their $cost$. The $cost$, in this regard, is based on a $cost$ function as in Equation 1 that takes as input the complexity of the CM ($\mathcal{O}()$) in terms of, e.g., the needed computational power and the introduced communication overhead. The $cost$ function also takes the financial cost (FC) of implementing a CM into account. In particular, implementing a specific CM, i.e., a PET, may require either hiring or consulting an expert in the field of that CM. Another aspect the $cost$ function should consider is the time (τ) required to implement that CM as implementing a

CM may substantially differ from implementing another. The factors the $cost$ function depends on are flexible and left to the privacy engineers to add, evaluate, and assess the importance of each factor, i.e., $x_1, ..., x_n$.

$$cost = f(\mathcal{O}(), FC, \tau, x_1, ..., x_n) \quad (1)$$

Algorithm 1: Toward selecting the most suitable CM to solve one privacy threat.

```
Input: Set of potential CMs to solve one threat
Output: The most suitable CM of the set
Ascendingly sort CMs by cost;
for CM in Set do
    if integrable and compatible then
        The most suitable CM in this set = this;
        integrate;
        endfor;
    end
end
if The most suitable CM in this set = null then
    modify or change system design;
end
```

Thereafter, each CM from the sorted list is checked whether it is integrable in the environment of the system, and compatible with other components, or not. Privacy engineers are thus able to select the CM that is 1.) integrable in the system, 2.) compatible with the other CMs used in the system, and 3.) intuitively, provides the minimum $cost$. Depending on the context, privacy engineers may also select a CM with a higher $cost$ if it, e.g., remedies several threats or aligns with the future plans of the developing organization. If none of the CMs in the sorted list is integrable nor compatible, then the current design of the system is to be changed so that a CM can be used to remedy the threat.

V. Future Work

The work presented in this paper opens the door to several aspects to be further investigated and analyzed. Future work should investigate to which end software engineers find the W-model applicable, and how easily the cost function can be estimated. Further investigation of the proposed CM-selection algorithm is required taking into account the scenario that a CM remedies several threats at once. Furthermore, future work should address formalizing the processes of checking both integrability and compatibility of a CM into system's context.

VI. Conclusion

In this paper, we discussed how to simplify the integration of privacy-related aspects into system development. We proposed the W-model as a privacy-aware variant of the common V-model for system development. Furthermore, we provided critical discussion on how to enable practitioners to more easily find solutions to elicited privacy threats. With the discussion provided in this paper, we aim to abate the confusion of selecting the most suitable countermeasure to remedy a privacy threat. Finally, we pointed out that there is great potential for future work to continue and build on what we proposed.

References

[1] GDPR — privacy by design. Last Checked: Feb. 2019. [Online]. Available: https://gdpr-info.eu/issues/privacy-by-design/

[2] S. Gürses, C. Troncoso, and C. Diaz, "Engineering privacy by design," 2011.

[3] S. Gurses, C. Troncoso, and C. Diaz, "Engineering privacy by design reloaded," in *Amsterdam Privacy Conference*, 2015, pp. 1–21.

[4] N. Notario, A. Crespo, Y.-S. Martin, J. M. Del Alamo, D. Le Métayer, T. Antignac, A. Kung, I. Kroener, and D. Wright, "PRIPARE: integrating privacy best practices into a privacy engineering methodology," in *Security and Privacy Workshops (SPW), 2015 IEEE*. IEEE, 2015, pp. 151–158.

[5] G. Danezis, J. Domingo-Ferrer, M. Hansen, J.-H. Hoepman, D. L. Metayer, R. Tirtea, and S. Schiffner, "Privacy and data protection by design-from policy to engineering," *arXiv preprint arXiv:1501.03726*, 2015.

[6] S. Spiekermann, "The challenges of privacy by design," *Communications of the ACM*, vol. 55, no. 7, pp. 38–40, 2012.

[7] M. Alshammari and A. Simpson, "Towards a principled approach for engineering privacy by design," in *Annual Privacy Forum*. Springer, 2017, pp. 161–177.

[8] S. Gürses, "Can you engineer privacy?" *Communications of the ACM*, vol. 57, no. 8, pp. 20–23, 2014.

[9] S. Gürses and J. Van Hoboken, "Privacy after the agile turn," *Cambridge Handbook of Consumer Privacy. Cambridge University Press, Cambridge*, 2017.

[10] M. Kuhrmann, D. Niebuhr, and A. Rausch, "Application of the V-Model xt–report from a pilot project," in *Software Process Workshop*. Springer, 2005, pp. 463–473.

[11] R. Rana, M. Staron, C. Berger, J. Hansson, M. Nilsson, and F. Törner, "Increasing efficiency of iso 26262 verification and validation by combining fault injection and mutation testing with model based development." in *ICSOFT*, 2013, pp. 251–257.

[12] M. Broy, "Challenges in automotive software engineering," in *Proceedings of the 28th international conference on Software engineering*. ACM, 2006, pp. 33–42.

[13] M. Deng, K. Wuyts, R. Scandariato, B. Preneel, and W. Joosen, "A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements," *Requirements Engineering*, vol. 16, no. 1, pp. 3–32, Mar 2011.

[14] L. Sion, K. Wuyts, K. Yskout, D. Van Landuyt, and W. Joosen, "Interaction-based privacy threat elicitation," in *International Workshop on Privacy Engineering*, 2018, pp. 2018–27.

[15] L. Sion, D. Van Landuyt, K. Yskout, and W. Joosen, "Sparta: Security & privacy architecture through risk-driven threat assessment," in *IEEE 2018 International Conference on Software Architecture (ICSA 2018)*. IEEE, 2018.

[16] L. Sion, K. Yskout, D. Van Landuyt, and W. Joosen, "Poster: Knowledge-enriched security and privacy threat modeling," in *2018 IEEE/ACM 40th International Conference on Software Engineering Companion (ICSE-C)*. IEEE/ACM, 2018.

[17] T. Antignac, R. Scandariato, and G. Schneider, "A privacy-aware conceptual model for handling personal data," in *International Symposium on Leveraging Applications of Formal Methods*. Springer, 2016, pp. 942–957.

[18] R. Galvez and S. Gurses, "The odyssey: modeling privacy threats in a brave new world," in *2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 2018, pp. 87–94.

[19] A. Kung, "PEARs: privacy enhancing architectures," in *Annual Privacy Forum*. Springer, 2014, pp. 18–29.

[20] K. Forsberg and H. Mooz, "The relationship of system engineering to the project cycle," in *INCOSE International Symposium*, vol. 1, no. 1. Wiley Online Library, 1991, pp. 57–65.

[21] S. Gürses and J. M. del Alamo, "Privacy engineering: Shaping an emerging field of research and practice," *IEEE Security & Privacy*, vol. 14, no. 2, pp. 40–46, 2016.