# Interaction-based Privacy Threat Elicitation

Laurens Sion, Kim Wuyts, Koen Yskout, Dimitri Van Landuyt, Wouter Joosen
*imec-DistriNet, KU Leuven*
Heverlee, Belgium
{laurens.sion, kim.wuyts, koen.yskout, dimitri.vanlanduyt, wouter.joosen}@cs.kuleuven.be

***Abstract*—Threat modeling involves the systematic identification, elicitation, and analysis of privacy- and/or security-related threats in the context of a specific system. These modeling practices are performed at a specific level of architectural abstraction – the use of Data Flow Diagram (DFD) models, for example, is common in this context.**

**To identify and elicit threats, two fundamentally different approaches can be taken: (1) elicitation on a per-element basis involves iteratively singling out individual architectural elements and considering the applicable threats, (2) elicitation at the level of system interactions (which involve the local context of three elements: a source, a data flow, and a destination) performs elicitation at the basis of system-level communication. Although not considering the local context of the element under investigation makes the former approach easier to adopt and use for human analysts, this approach also leads to threat duplication and redundancy, relies more extensively on implicit analyst expertise, and requires more manual effort.**

**In this paper, we provide a detailed analysis of these issues with element-based threat elicitation in the context of LINDDUN, an element-driven privacy-by-design threat modeling methodology. Subsequently, we present a LINDDUN extension that implements interaction-based privacy threat elicitation and we provide in-depth argumentation on how this approach leads to better process guidance and more concrete interpretation of privacy threat types, ultimately requiring less effort and expertise.**

**A third standalone contribution of this work is a catalog of realistic and illustrative LINDDUN privacy threats, which in turn facilitates practical threat elicitation using LINDDUN.**

## I. Introduction

Privacy by Design (PbD) is increasingly being recognized as an important and effective approach towards realizing privacy-preserving software solutions. This is confirmed by the introduction of the EU-wide General Data Protection Regulation (GDPR) [1], which applies to all systems that involve the processing of personal data, and mandates a risk-based approach to determine appropriate technical and organizational measures [1, e.g., art. 24]. Depending on the risk involved, a data protection impact assessment (DPIA) may be required as well [1, art. 35].

Threat modeling methodologies form an important class of solutions for conducting such analyses and assessments, as they enable a detailed and systematic analysis of the system under consideration. These modeling practices are performed at a specific level of architectural abstraction. For example, the STRIDE [2] threat modeling methodology for security, and LINDDUN [3], [4] which focuses exclusively on privacy, both start from a Data Flow Diagram (DFD) [5] that represents the main data-centric activities (processing, storage, and disclosure by/to external entities) in a system.

There is no clear consensus on what is the most appropriate level of abstraction for conducting threat elicitation. Originally, threat modeling approaches implemented an *element-based* approach, in which threat elicitation is performed by exhaustively iterating over individual architectural elements [3], [6], [7] (for example in a DFD context, iterating over all elements of the model individually). More recent evolutions however implement an *interaction-based* approach in which threat elicitation is done through systematic iteration over architecture-level interactions [2] (for example in DFDs, by iterating over 'source, data flow, destination'-combinations). Threat modeling easily leads to the problematic situation of *threat explosion* [8], [9] in which too many (less relevant) threats are raised. This negatively affects the overall cost-effectiveness of threat modeling practices. In this context, lack of consensus on what is the most effective threat elicitation approach is highly problematic.

As the first contribution of this article, we provide a detailed and in-depth analysis of the shortcomings of element-based threat elicitation, and we collect arguments on how interaction-based threat elicitation may resolve these issues. This is done through the creation of a variant of the LINDDUN [3] privacy threat modeling framework (in which threat elicitation is currently element-based) that performs interaction-based threat elicitation, and systematically comparing both variants. As this exercise involved detailed refinements and clarifications of key privacy-related terminology and threat types, the ensuing interaction-based variant of LINDDUN represents the second contribution of this article. As a third contribution, we provide a LINDDUN interaction-based table that provides illustrative privacy threats for all valid interactions in the context of DFDs, which serves as a key knowledge element to guide LINDDUN threat modelers in terms of interpretation and usage.

The remainder of this article is structured as follows. Section II introduces the necessary background on the LINDDUN threat modeling methodology and motivates the paper. Section III subsequently discusses the key issues with element-based threat elicitation. Section IV then introduces the interaction-based variant of LINDDUN, and introduces the example privacy threats for each valid DFD element combination, Section V provides a qualitative assessment of the benefits of interaction-based threat elicitation, after which Section VI gives a detailed discussion on lessons learned and future work. Section VII discusses related work and Section VIII concludes the paper.


https://doi.org/10.1109/EuroSPW.2018.00017

## II. Background and Motivation

This section provides background on the LINDDUN privacy by design framework, which is element-based in nature, and then provides a motivation for the paper by contrasting element-based threat elicitation approaches with interaction-based approaches.

### A. The LINDDUN privacy framework

LINDDUN [3], [10] is a state-of-the-art privacy threat modeling framework that provides support for systematic elicitation and mitigation of privacy threats (i.e., threats to the fundamental rights of data subjects) in software systems. Its main strength is the combination of a structured, methodological approach with an extensive privacy knowledge base consisting of threat trees that are structured according to the LINDDUN threat types.

*LINDDUN threat types:* LINDDUN is an acronym for the privacy threat types it investigates and supports. A short description for each of the high-level LINDDUN threat types is provided below.[1]

**Linkability** An adversary is able to link two items of interest without knowing the identity of the data subject(s) involved.

**Identifiability** An adversary is able to identify a data subject from a set of data subjects through an item of interest.

**Non-repudiation** The data subject is unable to deny a claim (e.g., having performed an action, or sent a request).

**Detectability** An adversary is able to distinguish whether an item of interest about a data subject exists or not, regardless of being able to read the contents itself.

**Disclosure of Information** An adversary is able to learn the content of an item of interest about a data subject.

**Unawareness** The data subject is unaware of the collection, processing, storage, or sharing activities (and corresponding purposes) of the data subject's personal data.

**Non-compliance** The processing, storage, or handling of personal data is not compliant with legislation, regulation, and/or policy.

*LINDDUN methodology:* LINDDUN is inspired upon STRIDE [11] and roughly shares the same methodological steps that guide the threat modeler from a *problem analysis* phase, in which threats are identified, to a *solution-oriented* phase, in which the identified threats are addressed systematically, i.e. from the selection of architecture-level mitigation strategies down to selection of specific Privacy-Enhancing Technologies (PETs) and architectural privacy patterns.

This article focuses mainly on the initial three steps of the LINDDUN methodology:

1) *Model the system.* The analysis of the software system starts from the creation of an architectural abstraction. In LINDDUN, the system is first modeled as a Data Flow Diagram (DFD) to capture the arguably most relevant system knowledge for conducting a privacy assessment: the data flows. The DFD notation is based upon 5 distinct building blocks: an *external entity* (i.e., users or third party services external to the system), a *process* (i.e., a unit of computation), a *data store* (i.e., a passive container of information), *data flow* (detailing how the data propagates through the system), and the *trust boundary* (i.e., a logical or physical division of the system). The example DFD model in Figure 1 is a simplistic representation of a social network, in which an external entity, `UserA`, connects through a web portal (the `Portal` process) to the (back-end) social network service (the `Service` process) and gains access to the information in the social network data store (the `Social network DB`).

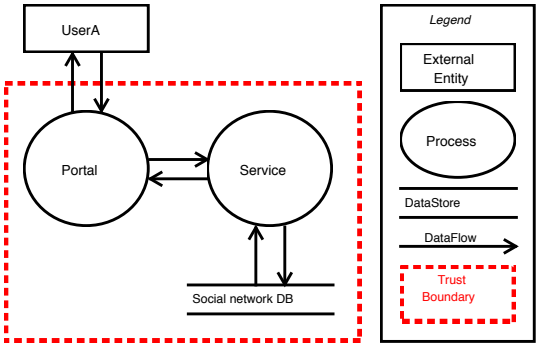



Fig. 1. Social Network DFD (from Deng et al. [3]).

TABLE I
Element-based LINDDUN mapping table template.

| Element Type | L | I | N | D | D | U | N |
|---|---|---|---|---|---|---|---|
| Process | X | X | X | X | X | - | X |
| DataStore | X | X | X | X | X | - | X |
| ExternalEntity | X | X | - | - | - | X | - |
| DataFlow | X | X | X | X | X | - | X |

2) *Map the LINDDUN threat types to the model.* As not all threat types apply to all DFD element types, the LINDDUN mapping template depicted in Table I highlights the combinations of LINDDUN threat types and DFD element types that are actually susceptible to privacy threats of these types (represented with an 'X'). By instantiating this template with the elements from the DFD of the system under investigation, a specific $N$x7 mapping table is created (with $N$ the amount of DFD elements in the model, and one column for each of the 7 LINDDUN threat types).

3) *Elicit and document privacy threats.* In the element-based elication approach that is taken in the current version of LINDDUN, for each individual 'X' in the mapping table, the corresponding DFD element has to be examined for the specific LINDDUN threat type. This leads to the identification and documentation of one or more[2] privacy-related threat(s). To provide guidance to the threat modeler, LINDDUN provides a set of threat trees [4] which describe (for each 'X' in

[1]For a more detailed description of LINDDUN, we refer to Wuyts et al. [4] and the LINDDUN website at https://linddun.org/.

[2]A single cell in the mapping table can correspond with multiple threats or even threat subtypes of that specific threat type.

the mapping template) the most common types of threats. Analyzing the elements in combination with the trees ensures that relevant threats are elicited.

### B. Element-based versus iteraction-based threat elicitation

As mentioned in Section I, LINDDUN implements an *element-based* elicitation approach, in which architectural elements are considered in isolation. While this represents a user-friendly approach for human threat modelers, there are some drawbacks.

*Interaction-based* threat elicitation approaches perform threat elicitation not at the basis of individual architectural elements, but at the basis of architecture-level interactions. In the context of DFDs, these are represented by data flows, i.e. 'source, data flow, destination'-combinations (e.g., `UserA`, data flow, `Portal` in Figure 1). As this allows the specific local context in the architectural abstraction to be taken into account, threats elicited through this approach are expected to be more explicit and precise about what they entail, and thus they are claimed to become more specific and easier to understand [2, p. 80].

An additional argument is that threats are seldom caused by individual architectural elements, but through data-centric interactions in the system [2], which inherently makes them a better subject for threat elicitation.

Finally, it is argued that since the amount of interactions will be less than the total amount of individual elements, less iterations are required. While the number of threats may stay the same or even increase, the number of context switches that the threat modeler has to make is reduced.

Existing and readily-available tool support for security threat modeling, such as the Microsoft Threat Modeling Tool [12], already successfully employs interaction-based expressions as the basis for eliciting security threats.

Although interaction-based threat elicitation was initially introduced for the purposes of reducing the amount of threats to be elicited —and thus to increase the overall cost-effectiveness— some reports highlight that it actually causes the elicitation of more threats [2].

**Motivation.** This overall lack of consensus on what is the most appropriate basis for threat elicitation is problematic. This issue is of high relevance in the research towards making threat modeling approaches more cost-efficient and practical for adoption.

## III. Analysis of element-based threat elicitation

This section illustrates and discusses the main benefits and issues with the application of element-based threat elicitation, while referring to the example social network DFD presented in Figure 1 and the element-based LINDDUN threat type mapping template shown in Table I.

The main benefit of the element-based approach is its inherent simplicity. There is a single, small mapping template (shown in Table I) that guides the elicitation process. As discussed in Section II-A, the mapping table details for each DFD element type, a row of 'X's and '—'es that indicate which LINDDUN threat types are applicable for which DFD element type. Adopting the element-based threat elicitation approach involves simply iterating over each element of the DFD model under analysis, and for each element identifying and documenting threats (of the applicable threat types) according to the template. This is particularly suited for manual execution performed by human threat modelers.

As mentioned, the lack of local context information during threat elicitation may lead to (i) omission of relevant threats (undiscovered threats, false negatives), (ii) the elicitation of threats that are not applicable (false positives), and (iii) redundancy in documenting threats. We discuss and illustrate each of these problems below.

**1. Undiscovered threats.** The element-based threat elicitation approach can lead to the omission of several applicable threats for elements involved in more than a single interaction. When there are multiple incoming or outgoing *data flows* on an element, the element may be threatened in different ways over each of those flows. By not taking these interactions into account, but instead only focusing on the element itself, some of those threats may remain undiscovered as the context of these different interactions is unknown by the analyst.

For example, insufficiently minimizing results from the DB can cause *Identifiability* issues, i.e., it remains possible to read or derive the identity of the data subject. But also, queries to the DB can reveal information about the user that is performing the query itself. Therefore, not considering the additional roles the DB plays in the other interactions can cause the threat modeler the miss the second *Identifiability* issue that may affect all the other interactions in which the data store is involved.

**2. Eliciting threats that are not applicable.** Not all threat types always apply, and the applicability of a threat type depends highly on the direction of the data flow. For example, a *Detectability* threat at the level of a process is only applicable when the process is sending data, not when receiving. An 'X' for *Detectability* at a *process* in the threat mapping template may cause much wasted analysis effort in trying to elicit a threat that is not applicable in the first place. By considering the applicability of privacy threat types on a more fine-grained basis, such types can be eliminated beforehand and do not need to be considered further.

Consider, for example, the elicitation of privacy threats at the level of the `Social network DB` data store shown at the bottom of Figure 1. A *Detectability* threat at the level of the data store on records of the users is only applicable if there actually is a flow from the data store to the `Service` process. If the data store does not respond to any request (not even reporting a write success or failure), then it simply cannot reveal the presence of existing records, and there is no plausibile *Detectability* issue. Elicitation of threats without taking such context into account in this example leads to the identification and documentation of a *Detectability* threat that will have to be removed in later steps.

**3. Eliciting redundant threats.** Observations on the application of LINDDUN's security counterpart, STRIDE, in an industrial context, indicate that the element-based threat

approach is more time-consuming and redundant [13, p. 43]. Without taking into account the interaction and its directions, threat elicitation can lead to duplicate threat elicitations (at the source, the data flow and the destination). This is caused by the lack of a clear distinction between these different roles involved in a single interaction.

## IV. LINDDUN by Interaction

This section introduces the interaction-based variant of LINDDUN. First, Section IV-A outlines the adopted approach for creating the corresponding mapping table template. Then, Section IV-B introduces the LINDDUN by interaction variant. Finally, Section IV-C presents an illustrative mapping table that can help threat threat modelers to adopt this more extensive version of LINDDUN.

### A. Construction of the interaction-based threat mapping template

Table II presents the threat type mapping template for interaction-based LINDDUN.

The creation of this template started with listing all possible combinations of DFD element interactions (excluding invalid DFD combinations such as flows between data stores), taking into account that each interaction has three different elements which can be focused on (i.e., source, data flow or destination), leading to a total of 105 table cells.

As the interpretation of the privacy threat types varies slightly compared to the element-based approach, and to avoid interpretation ambiguities, we started from a completely empty mapping template table and systematically evaluated for each privacy threat type whether it would be applicable in the specific context of the data flow (and the participating element types) under investigation. This initial table was created at the basis of expert knowledge (with the involvement of the creators of LINDDUN) and guided by illustrative examples (see Section IV-C).

After this, we compared the resulting table with the original LINDDUN table by manually performing an in-depth delta analysis. This exercise highlighted several differences (5 additions and 12 eliminations of 'X's, or 16% of the entries). These discrepancies mainly originated from a lack of semantics for associating a threat to a specific participating element in the context of a data flow. Therefore, the following interpretations were agreed upon and used to unambiguously specify whether a privacy threat should be linked to the *source*, the *data flow*, or the *destination*:

**Source** The threat arises at the level of the element that shares or communicates data where the sharing of the data can cause a privacy threat. (E.g., a browser that retransmits cookies or other linkable identifiers to each recipient.)

**Data Flow** The threat arises at the level of the data flow, i.e., when the data (both contextual, i.e., meta-data, and transactional, i.e., the content itself, are in transit. (E.g., contextual data about the source and destination can be used to link multiple data flows, or to identify the parties involved in the communication.)

TABLE II
INTERACTION-BASED LINDDUN THREAT TYPE MAPPING

| Source | | Destination | L | I | N | D | D | U | N |
|---|---|---|---|---|---|---|---|---|---|
| ***Process*** | flow | Process | X | X | X | X | X | – | X |
| ***Process*** | flow | DataStore | X | X | X | X | X | – | X |
| ***Process*** | flow | ExternalEntity | X | X | X | X | X | – | X |
| ***DataStore*** | flow | Process | X | X | X | X | X | – | X |
| ***ExternalEntity*** | flow | Process | X | X | X | X | X | X | – |
| Process | flow | ***Process*** | X | X | X | – | – | – | X |
| DataStore | flow | ***Process*** | X | X | X | – | – | – | X |
| ExternalEntity | flow | ***Process*** | X | X | X | – | – | – | X |
| Process | flow | ***DataStore*** | X | X | X | – | X | – | X |
| Process | flow | ***ExternalEntity*** | X | X | X | – | – | X | X |
| Process | ***flow*** | Process | X | X | X | X | X | – | – |
| Process | ***flow*** | DataStore | X | X | X | X | X | – | – |
| Process | ***flow*** | ExternalEntity | X | X | X | X | X | – | – |
| DataStore | ***flow*** | Process | X | X | X | X | X | – | – |
| ExternalEntity | ***flow*** | Process | X | X | X | X | X | – | – |

The elements in the first three columns highlight the element to which the privacy threat is associated (using a ***colored and emphasized*** notation). Note that invalid DFD element combinations (such as *DataStore-flow-DataStore* or *ExternalEntity-flow-ExternalEntity*) are not included in this table.

**Destination** The threat arises at the level of the element that receives the data where the data can be processed or stored in a way that causes a privacy threat. (E.g., insecure storage or insufficient minimization of the data upon storing.)

### B. Using the interaction-based threat mapping template

In terms of elicitation approach, Figure 2 illustrates the main differences between element-based and interaction-based elicitation over a small subset of the social network DFD: instead of iterating over each DFD element, the interaction-based approach involves iterating over data flows (as the iteration over each such interaction also involves the elicitation of the threats at the source, the data flow or the destination). Consequently, each investigated interaction allows different threats to be specified for the three participants involved in the interaction. Each of these privacy threat types correspond to one of the three main outer rows in Table II, in which the first three columns mark the element type to which the threat is associated.

### C. LINDDUN by interaction: illustrative table

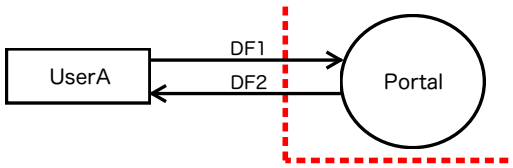
Finally, Table III presents a full table of illustrative and concrete privacy threats for all possible valid combinations of DFD element types and privacy threat types. This table will aid the human threat modeler with understanding and interpreting the semantics of the different threat types and as such serves as a key knowledge asset for brainstorming about potential privacy threats during the elicitation phase.

Obviously, such a table of examples is inherently incomplete, as multiple privacy threat (sub-)types may apply for a single cell. To find all potential privacy threat types, additional resources such as privacy threat trees [10] can be used in conjunction with this table.

TABLE III
INTERACTION-BASED LINDDUN EXAMPLES

| Source | Destination | Linkability | Identifiability | Non-Repudiation | Detectability | Information Disclosure | Unawareness | Non-Compliance |
|---|---|---|---|---|---|---|---|---|
| ***Process*** | Process | Cookies sent by the browser (S) can link a data-subject's activities. | A browser's (S) eID login to a website (D) leaks identity info | Sender (S) has no plausible deniability without precautions (e.g., OTR chat). | Website (S) showing incorrect password error reveals account existence. | Process (S) reveals data to unauthorized recipient (D). | – | Processing (S) user data without lawful basis. |
| ***Process*** | DataStore | Process (S) uses identifiers (e.g., certificate) or predictable timing. | Identifiable information in source's (S) eID certificate (i.e., full name, etc.). | Sender (S) has no plausible deniability without precautions (e.g., OTR chat). | Website (S) showing incorrect password error reveals account existence. | Writing (S) data to DB (D) which shouldn't be there (e.g., passwords). | – | Processing (S) user data without lawful basis. |
| ***Process*** | – *flow* – External-Entity | Sharing (S) email addresses can allow a 3rd party (D) to link it with own data. | Sharing (S) identifiable info enables 3rd party (D) data-subject identification. | Sender (S) has no plausible deniability without precautions (e.g., OTR chat). | Communicating (S) PII to 3rd parties (D), enables detection of your users. | 3rd party (D) receives data from process (S) it's not authorized to receive. | – | Sharing (S) with a 3rd party (D) without lawful basis. |
| ***DataStore*** | Process | Insufficiently anonymized data in DB (S) can be linked (D). | Insufficiently anonymized data in DB (S) could be de-anonymized (D). | Sender (S) has no plausible deniability without precautions (e.g., OTR chat). | Write errors from DB (S) can reveal that a record already exists to (D). | A database (S) without access control can disclose information to (D). | – | DB (S) Stores or sends data without lawful basis. |
| ***External-Entity*** | Process | Sending (S) pseudonymous identifiers can lead to linkability (D). | Sending (S) identifiable data to (D). | Sender (S) has no plausible deniability without precautions (e.g., OTR chat). | Context data from sensors (e.g., accelerometer) at (S) can be used for detection (D). | Process (D) is not authorized to receive the data from (S). | Data-subject (S) is unaware of which or how much data is being shared with (D). | – |
| Process | ***Process*** | An SSO service provider (D) can link users' (S) movements across services. | Sign-up via SSO (D) can lead to identifiability if the SSO sends user data. | A process (D) adding digital signatures. | – | – | – | Processing (D) user data without lawful basis. |
| DataStore | ***Process*** | A browser process (D) links all history (S) in single user session. | Retrieval (D) of identifiable information from DB (S). | A process (D) adding digital signatures. | – | – | – | Processing (D) user data without lawful basis. |
| External-Entity | – *flow* – ***Process*** | An SSO service provider (D) can link users' (S) movements across services. | Sign-up via SSO (D) can lead to identifiability if the SSO sends user data. | A process (D) adding digital signatures. | – | – | – | Processing (D) user data without lawful basis. |
| Process | ***DataStore*** | Insufficient data minimization or anonymization (D). | Queries to a DB (D) can reveal info about the user (S) performing them. | Logging (D) user (S) actions prevents plausible deniability. | – | Insecure storage (e.g., unencrypted) (D). | – | Insufficient minimization/ anonymization (D) |
| Process | ***External-Entity*** | Sending (S) identifiers to 3rd party (D) allows it to combine it with its own info. | Sending (S) identifiable info to a 3rd party (D) enables identification. | Sending (S) user interaction records to a 3rd party (D) prevents plausible deniability. | – | – | Data-subject is unaware of sharing of data with 3rd parties (D). | Sharing data with a 3rd party which is non-compliant. |
| Any | – ***flow*** – Any | Linking flows using the meta-data. | Identifiable data in an unprotected data flow. | Capture data flow to collect evidence of communication. | Detection of data flows (e.g., wifi traffic). | Data-flow contents and meta-data are sniffed on the wire. | – | – |

The elements in the first three columns highlight the element to which the privacy threat is associated (using a ***colored and emphasized*** notation), using the definitions from Section IV-A. The (S) and (D) annotations in the examples refer to the source and destination elements respectively.




| Element-based | Interaction-based |
|---|---|
| **Iteration 1:** Data Flow DF1 | |
| → DF1 threat(s) | → DF1 threat(s) |
| - | → UserA threat(s) (*source*) |
| - | → Portal threat(s) (*destination*) |
| **Iteration 2:** Data Flow DF2 | |
| → DF2 threat(s) | → DF2 threat(s) |
| - | → UserA threat(s) (*destination*) |
| - | → Portal threat(s) (*source*) |
| **Iteration 3:** External Entity UserA | |
| → UserA threat(s) | - |
| **Iteration 4:** Process Portal | |
| → Portal threat(s) | - |

Fig. 2. Illustration of the differences between the element-based and interaction-based iterations.

## V. Qualitative Assessment

We provide a qualitative assessment of the interaction-based LINDDUN approach through direct comparison with element-based LINDDUN. The following four aspects are assessed: (i) the *expressivity* of the interaction-based approach; (ii) the a-priori *elimination* of non-applicable privacy threat types; (iii) the prevention of *undiscovered* threats; and (iv) the *effort-precision* trade-off.

For each of the qualitative claims made during this assessment, a more in-depth empirical validation is part of our future work.

**Expressivity** First, we assess whether the interaction-based table is at least as expressive as the element-based table, i.e., there are no threats that can be elicited in the element-based variant that could not be elicited with the interaction-based variant. This is addressed explicitly during construction of the interaction-based mapping table, i.e., in the third step (delta analysis) as discussed in Section IV-A, in which we explicitly verified that the element-based table can be completely captured in the interaction-based one.

Additionally, in contrast to the element-based approach, the interaction-based template can be used to express semantical differences between threats associated to source, data flow or destination elements. This is illustrated in Table II through the differences between the three main outer rows. The semantic distinction between, for example, a process as source or destination for *Detectability*, can not be expressed in the element-based threat mapping template because there is only a single row for the element regardless of the element's role in an interaction. We argue that this leads to the identification of more precise and fine-grained threats, as the information on the element's role can now explicitly be taken into account.

**Elimination of non-applicable privacy threat types** *Detectability* is an example of a privacy threat type which is applicable for a source element but not a destination element. This consideration is encoded in the mapping table template, whereas in the element-based threat elicitation approach, this distinction cannot be made. This could lead the threat modeler to waste much effort in looking for a *Detectability* threat on an element that only acts as the destination of a flow, for example.

We argue that eliminating threat types that are inherently not applicable in the specific context of an interaction up front improves the overall cost-efficiency of threat elicitation.

**Undiscovered threats** By focusing only on one type of role (e.g., source) during the element-based threat elicitation, certain threats easily remain undiscovered. The interaction-based elicitation forces the analyst to systematically consider each element in its specific roles, which inherently ensures the inclusion of these types of threats. We argue that using a more fine-grained level of granularity leads to an improved recall of applicable threats.

**Effort-precision trade-off** Instead of specifying privacy threat types at the level of a DFD element, the interaction-based approach specifies them at the level of a 'source-data flow-destination' combination in which the threat is located at one of the elements. In the example DFD, shown in Figure 1, the total number of required iterations during threat elicitation changes from 10 (the amount of DFD elements) to 6 (the amount of data flows). For each of those 6 iteration steps, threats have to be considered from the point of view of (at most) 3 distinct roles (source, data flow, and destination), depending on the applicability of the threat type in the mapping table.

While this does come with an increase in effort per iteration, we argue that the interaction-based approach provides an efficiency gain because of the expertise that is encoded and reused to determine a privacy threat type's applicability at a fine-grained level.

Applying interaction-based threat elicitation involves an inherent trade-off between the time/effort to be spent during analysis, and the overall precision and efficiency of threat elicitation. This trade-off has also been evaluated in the context of security threat modeling. Dhillon [13] described that such a trade-off has been positively perceived in the application of STRIDE security threat modeling at EMC.

## VI. Discussion and Future Work

As discussed in the previous section, the application of interaction-based threat modeling addresses a number of the issues with element-based threat elicitation. In this section, we provide further discussion in terms of lessons learned and future work, covering: (i) semantics and ambiguities of privacy threats; (ii) the role of other knowledge elements such as threat trees; (iii) next steps towards tool support; and (iv) whether or not the most appropriate basis for threat elicitation will be the same architectural abstraction, regardless of the nature of the threat type under investigation.

**Semantics and ambiguities of privacy threats** As discussed in Section IV-A, the construction of the mapping table template was accompanied with several discussions on the semantics of privacy threats and, more specifically, on the ambiguities in the interpretation of what associating a specific privacy threat type to a specific element type entails. Most of

these ambiguities stem from varying interpretations in terms of where a threat manifests itself, the target of a threat (what/who is threatened), and the convolution of both.

These properties of privacy threats are quite tricky as they differ fundamentally from the more traditional interpretations in the context of security threat modeling [2]. Security threat modeling is commonly performed from the point of view of deliberate and malicious adversaries. For example, the element at the other end of an interaction can be malicious and pose a active threat to the first (threatened) element.

A key distinguishing factor of privacy threats is that the threatened entity is in all cases (the data protection rights of) one (or more) data subject(s), i.e., the person(s) whose data is processed by the system. As such, the threatened data subject does not have to be explicitly involved in the interaction for a privacy threat to occur. For example, a database containing incorrectly or insufficiently anonymized data that is shared with third parties poses a privacy threat. Furthermore, all the participating elements in a single interaction can actually be malicious, in the sense that they all contribute in realizing a privacy threat to the data subject, without even being controlled or subverted by an active adversary.

The exercise of constructing the mapping table template (as discussed in Section IV-A) was instrumental in finding expert consensus on the precise semantics of privacy threats in the context of Data Flow Diagrams, and provides clear pointers for future work. For example, providing the DFD modeler with the means to concretize the nature of the processed data (in terms of sensitivity) or the nature of the involved data subject (e.g., by providing meta-data in the DFD) is one promising direction for further improvements.

**Threat trees** As discussed, interaction-based threat elicitation approaches allow for a more fine-grained specification of privacy threat types. The knowledge resources that support a threat modeling approach are commonly structured in accordance to the nature of the supported threat elicitation approach. For example, LINDDUN threat trees [4] and STRIDE threat trees [2] are structured according to the element-based mapping table (one threat tree per 'X' in the mapping table). In future work, we envision restructuring these trees in alignment with the interaction-based approach, which we expect will facilitate the concretization and consideration of the currently implicit causality relations between different threat types and (sub-)types such as, for example, *Linkability* leading to *Identifiability*.

**Usage & tool support** Not only does the more fine-grained privacy threat type mapping template assist the human threat modeler in the elicitation of privacy threats, it can also be instrumental towards creating advanced tool support for more automated privacy threat elicitation.

The interaction-based approach has already been successfully implemented in tool support for the automatic elicitation of security threats [8], [12], indicating that a similar effort for automatic privacy threat elicitation would be fruitful. The implementation of such tool support for automatic privacy threat elicitation would then in itself further enable more advanced privacy threat analysis activities in order to obtain a comprehensive and tool-supported privacy analysis framework. This is part of our ongoing work.

**Granularity for Threat Elicitation** We have argued the extent to which the abstraction level of an *architectural interaction* is better suited than that of *individual elements* for eliciting LINDDUN privacy threats. In earlier threat analysis exercises, we have, however, encountered more sophisticated threats that do not fit easily in the confines of neither single elements nor interactions, for example, *Detectability* threats that are enacted by monitoring the end-to-end response time of a request. Arguably, such threats will not be discovered in a plain LINDDUN analysis. This raises the key question whether the most appropriate abstraction level for eliciting threats should depend on the inherent nature of the considered privacy threat type. This is especially relevant in our ongoing work towards automating and systematizing the elicitation logic. In this case, the formalization of the required context elements and conditions for determining the plausibility of a specific threat type is highly necessary.

## VII. Related Work

The broader field of privacy engineering focuses on systematically identifying and addressing privacy issues [14], [15], but also the development of design strategies and architectural patterns [16], privacy-enhancing technologies (PETs) and cryptographic enablers.

Methods and approaches that involve systematic threat modeling represent a good candidate for implementing the end-to-end risk-based privacy analysis demanded by the GDPR [1]. These methods can be considered the technical component (in addition to legal, organizational, and business-oriented measures) of the privacy impact assessment (PIA) which involves estimating and resolving overall privacy risks.

Microsoft [6], [17] originally introduced the STRIDE threat modeling approach as part of its security development life cycle (SDL) [7]. This approach in its origins involves element-based threat elicitation. More recently, STRIDE has evolved towards interaction-based threat elicitation [2], also in tool support [12]. STRIDE is actively used in industry, both within Microsoft [18] and elsewhere [13].

In addition to the LINDDUN methodology [3], [4] that was extended in this article, other initiatives to privacy by design are noteworthy:

PRIAM [19] is a framework for conducting a systematic privacy risk assessment, not at the level of technical architecture but at the level of data catalogs. It uses harm trees to link privacy weaknesses and risk sources to known harms. Oliver [20] proposes an approach based on data flow modeling that is enriched with ontologies and classifications to annotate and describe information flows. This approach is element-based in nature. Oetzel and Spiekermann [21] propose a step-by-step privacy impact assessment (PIA) starting from legal requirements (privacy targets), leading to a list of control recommendations. This approach does not systematically zoom into a technical architecture. Shapiro [22] introduced the

System-Theoretic Process Analysis for privacy engineering process (STPA-Priv), which is similar to STPA-Sec, an earlier adaptation of such approach focused on security [23]. These approaches do not involve eliciting threats at the level of a technical architecture, but at the more coarse grained-level of a blackbox system and its context. Finally, Shapiro [24] also introduced STECA-Priv which instantiates Systematic-Theory Early Concept Analysis in a privacy engineering domain. Contrary to the STPA-Priv or threat modeling methodologies, it does not yet require a system design, but instead starts from the system specification to infer a privacy control structure [24].

## VIII. Conclusion

The LINDDUN privacy threat modeling framework currently involves element-based threat elicitation, which involves iteration over all elements of a DFD model without taking any architectural context into account. In this article, we (i) argued why this is sub-optimal in the context of privacy threat elicitation, (ii) provided a usable extension to the LINDDUN privacy threat modeling framework that supports threat elicitation at the basis of interactions, and (iii) provided a set of concretizations and illustrative LINDDUN threat examples which are structured according to the more fine-grained per-interaction threat mapping template. This resulting table serves as a key knowledge asset for brainstorming about potential privacy threats during the elicitation phase.

This work fits into our ongoing research towards more streamlined tool support for LINDDUN through consolidating key privacy expertise in a more reusable form. This as such improves the overall cost-effectiveness of these techniques, which remains the main hurdle for broad adoption of threat modeling in general.

## Acknowledgment

This research is partially funded by the Research Fund KU Leuven and the PRiSE research project.

## References

[1] European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016," *Official Journal of the European Union*, vol. 59, no. L 119, pp. 1–88, may 2016.

[2] A. Shostack, *Threat Modeling: Designing for Security*. Indianapolis, Indiana: John Wiley & Sons, 2014.

[3] 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, 2011.

[4] K. Wuyts, "Privacy Threats in Software Architectures," Ph.D. dissertation, KU Leuven, jan 2015.

[5] T. DeMarco, *Structured Analysis and System Specification*. Yourdon Press, 1979.

[6] F. Swiderski and W. Snyder, *Threat modeling*. Microsoft Press, 2004.

[7] M. Howard and S. Lipner, *The Security Development Lifecycle*. Microsoft Press, 2006.

[8] L. Sion, K. Yskout, D. Van Landuyt, and W. Joosen, "Solution-aware Data Flow Diagrams for Security Threat Modelling," in *Proceedings of The 6th track on Software Architecture: Theory, Technology, and Applications*, 2018, p. (to appear).

[9] K. Wuyts, D. Van Landuyt, A. Hovsepyan, and W. Joosen, "Effective and Efficient Privacy Threat Modeling through Domain Refinements," in *Proceedings of The 1st track on Privacy by Design in Practice (SAC 2018)*, 2018, p. (to appear).

[10] K. Wuyts and W. Joosen, "LINDDUN privacy threat modeling: a tutorial," 2015.

[11] M. Howard and S. Lipner, *The security development lifecycle*. O'Reilly Media, Incorporated, 2009.

[12] Microsoft Corporation, "Microsoft Threat Modeling Tool 2016," http://aka.ms/tmt2016, 2016.

[13] D. Dhillon, "Developer-Driven Threat Modeling: Lessons Learned in the Trenches," *IEEE Security Privacy*, vol. 9, no. 4, pp. 41–47, jul 2011.

[14] S. Gürses, C. Gonzalez Troncoso, and C. Diaz, "Engineering privacy by design," in *Proceedings of the Conference on Computers, Privacy & Data Protection (CPDP 2011)*, 2011, p. 25.

[15] S. Gürses, C. Troncoso, and C. Diaz, "Engineering Privacy by Design Reloaded," *Amsterdam Privacy Conference 2015*, pp. 1–21, 2015.

[16] M. Colesky, J. H. Hoepman, and C. Hillen, "A Critical Analysis of Privacy Design Strategies," in *2016 IEEE Security and Privacy Workshops (SPW)*, may 2016, pp. 33–40.

[17] S. Hernan, S. Lambert, T. Ostwald, and A. Shostack, "Threat Modeling: Uncover Security Design Flaws Using The STRIDE Approach," *MSDN Magazine*, vol. 6, nov 2006.

[18] A. Shostack, "Experiences threat modeling at Microsoft," in *Modeling Security Workshop. Dept. of Computing, Lancaster University, UK*, 2008.

[19] S. Joyee De and D. Le Métayer, "PRIAM: A privacy risk analysis methodology," *Lecture Notes in Computer Science*, pp. 221–229, 2016.

[20] I. Oliver, *Privacy engineering: A dataflow and ontological approach*. CreateSpace Independent Publishing Platform, 2014.

[21] M. C. Oetzel and S. Spiekermann, "A systematic methodology for privacy impact assessments: A design science approach," *European Journal of Information Systems*, vol. 23, no. 2, pp. 126–150, 2014.

[22] S. S. Shapiro, "Privacy Risk Analysis Based on System Control Structures: Adapting System-Theoretic Process Analysis for Privacy Engineering," in *2016 IEEE Security and Privacy Workshops (SPW)*, may 2016, pp. 17–24.

[23] W. Young and N. G. Leveson, "An Integrated Approach to Safety and Security Based on Systems Theory," *Communications of the ACM*, vol. 57, no. 2, pp. 31–35, feb 2014.

[24] S. S. Shapiro, "Addressing Early Life Cycle Privacy Risk," in *2017 International Workshop on Privacy Engineering - IWPE'17*, 2017.