Noname manuscript No.
(will be inserted by the editor)

A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements

Mina Deng · Kim Wuyts · Riccardo Scandariato · Bart Preneel · Wouter Joosen

Received: June-02-2010 / Accepted: May-01-2010

Abstract Ready or not, the digitalization of information has come and privacy is standing out there, possibly at stake. Although digital privacy is an identified priority in our society, few systematic, effective methodologies exist that deal with privacy threats thoroughly. This paper presents a comprehensive framework to model privacy threats in software-based systems. First, this work provides a systematic methodology to model privacy-specific threats. Analogous to STRIDE, an information flow oriented model of the system is leveraged to guide the analysis and to provide broad coverage. The methodology instructs the analyst on what issues should be investigated, and where in the model those issues could emerge. This is achieved by (i) defining a list of privacy threat types and (ii) providing the mappings between threat types and the elements in the system model. Second, this work provides an extensive catalogue of privacy-specific threat tree patterns that can be used to detail the threat analysis outlined above. Finally, this work provides the means to map the existing privacy-enhancing technologies (PETs) to the identified privacy threats. Therefore, the selection of sound privacy countermeasures is simplified.

Keywords privacy · threat modeling · requirements · secure software engineering

Mina Deng, Bart Preneel
IBBT-COSIC, Electrical Engineering Depterment, K.U.Leuven
Kasteelpark Arenberg 10, B-3001 Heverlee, Belgium
E-mail: first.last@esat.kuleuven.be

Kim Wuyts, Riccardo Scandariato, Wouter Joosen
IBBT-DistriNet, Computer Science Department, K.U.Leuven
Celestijnenlaan 200A, B-3001 Heverlee, Belgium
E-mail: first.last@cs.kuleuven.be

1 Introduction

Privacy becomes increasingly important in the current society. Most of the information is now digitalized to facilitate quick and easy access. It is thus extremely important that digital privacy is sufficiently protected to prevent personal information from being revealed to unauthorized subjects. A stepping stone of security and privacy analysis is threat modeling, i.e., the "black hat" activity of looking into what can possibly go wrong in a system. Threats are crucial to the definition of the requirements and play a key role in the selection of the countermeasures. Unfortunately, the state of the art lacks systematic approaches to model privacy threats, elicit privacy requirements, and instantiate privacy-enhancing countermeasures, accordingly. Indeed, there is an asymmetry for privacy with respect to security concerns. These latter have a far better support in terms of methodological approaches to threat modeling. For instance, in the goal-oriented requirements space, KAOS [1] provides a methodology to systematically analyze a system's anti-goals (and the corresponding refined threats) and therefore derive security requirements [2]. The same holds in the area of scenario-based techniques. For instance, Microsoft's STRIDE is an industrial-level methodology to eliciting threat scenarios and, therefore, deriving security use cases [3]. Notably, a significantly sized body of reusable knowledge is also available in the secure software engineering community. Security knowledge is often packaged in the shape of checklists and patterns. For instance, STRIDE comes bundled with a catalogue of security threat tree patterns that can be readily instantiated in the system at hand so to elicit a close-to-exhaustive set of potential security threats. Methodologies and knowledge are two important pillars for software security and privacy, including requirements engineering [4]. Surprisingly, privacy is still lagging behind. For instance, STRIDE does not cover privacy threats.

This paper contributes to the aforementioned dimensions, in terms of methodology and knowledge, by providing a comprehensive privacy threat modeling framework. A high-level overview of this work is sketched out in Section 3.

First, this work provides a systematic methodology to model privacy-specific threats. Analogous to STRIDE, an information flow oriented model of the system is leveraged to guide the analysis and to provide broad coverage. The data flow diagram (DFD) notation has been selected and, for reference, it is described in Section 2. The methodology instructs the analyst on what issues should be investigated and where in the model those issues could emerge. This is achieved by defining a list of privacy threat types and by providing the mapping between the threat types and the elements in the system model. This part of the methodology is described in Section 5. Note that the privacy threat types

have been identified in contrast with well known privacy objectives, which are summarized in Section 4.

Second, this work provides an extensive catalogue of privacy-specific threat tree patterns that can be used to detail the threat analysis outlined above. In a nutshell, they refine the privacy threat types by providing concrete examples. The catalogue is described in Section 6, while Section 7 illustrates how to instantiate the threat tree patterns in order to elicit the misuse cases.

An additional contribution of this paper refers to the software engineering phase. This work provides the means to map the existing privacy-enhancing technologies (PETs) to the identified privacy threats, which simplifies the selection of sound privacy countermeasures. This is described in Section 8.

Concerning the rest of the paper, the related work is presented in Section 9, a discussion of the proposed methodology is presented in Section 10 and the concluding remarks are given in Section 11.

2 Background: security threat modeling using STRIDE

Security, in contrast to privacy, has already been well integrated in the Secure Development Lifecycle (SDL) [3], which is a well-established methodology. To build a secure software system, an important aspect is to consider how an attacker might compromise the system by exploiting design flaws and building the necessary defense mechanisms in the system. In this respect, threat modeling plays the key role, and SDL has integrated a systematic approach for security threat modeling using STRIDE. In this section, we will briefly review the STRIDE threat modeling process, which consists of nine high-level steps.

Step 1: Define use scenarios. System designers need to determine which key functionality is within the scope.

Step 2: Gather a list of external dependencies. Each application depends on the operating system it runs on, the database it uses, and so on; these dependencies need to be defined.

Step 3: Define security assumptions. In the analysis phase, decisions are often based on implicit assumptions. Therefore, it is important to note down all the assumptions, to understand the entire system comprehensively.

Step 4: Create external security notes. Because each external dependency can have its implication on security, it is useful to list all the restrictions and implications introduced by the external security notes. An example of such a security note is to specify which ports are open for database access or HTTP traffic.

Step 5: Create one or more DFDs of the application being analyzed. The software-based system being analyzed is decomposed in relevant (either logical or structural) components, and for each of these parts the corresponding threats are analyzed. This process is repeated over an increasingly refined model until a level is reached where the residual threats are acceptable.

The system is graphically represented using a data flow diagram (DFD), with the following elements: data flows (i.e. communication data), data stores (i.e. logical data or concrete databases, files, and so on), processes (i.e. units of functionality or programs) and external entities (i.e. endpoints of the system like users, external services, and so on). For threat modeling, trust boundaries are also introduced to indicate the border between trustworthy and untrustworthy elements.

An example DFD is shown in Figure 1 to illustrate a use case application (Social Network 2.0) that will be discussed throughout this paper. This Social Network 2.0 application is an abstract representation of a social network, where online users share personal information such as relationship status, pictures, and comments with their friends. In the DFD, the user is represented as an entity to interact with the system. The Social Network 2.0 application contains two processes (the portal and the service) and one data store containing all the personal information of the users. The trust boundary shows that the processes, the data store, and the communication (data flows)between the two are assumed to be trustworthy in this particular setting.

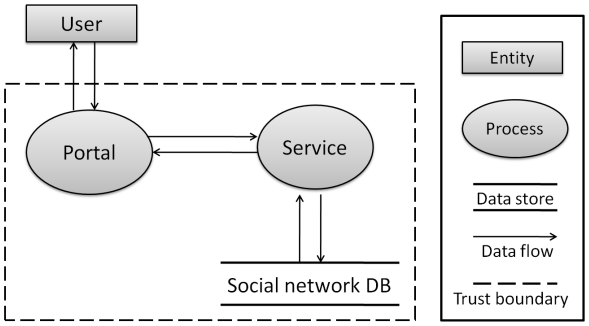

Fig. 1 The Data Flow Diagram (DFD) of the Social Network 2.0 application

Step 6: Determine threat types. The STRIDE threat taxonomy is used to identify security threat types. STRIDE is an acronym for Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privilege. These threats are the negation of the main security properties, namely confidentiality, integrity, availability, authentication, authorization and non-repudiation.

Step 7: Identify the threats to the system. Each element of the data flow diagram is assigned to a set of susceptible threats. Table 1 gives an overview of the different DFD elements with the corresponding security threats they are subject to (marked with $\times$).

Table 1 Security concerns with corresponding security threats and DFD elements susceptible to threats (DF-Data flow, DS-Data store, P-Process, E-External entity), proposed by the Security Development Lifecycle (SDL) [3].

Security property	Security threat	DF	DS	P	E
Authentication	Spoofing			×	×
Integrity	Tampering	×	×	×	
Non-repudiation	Repudiation		×	×	×
Confidentiality	Information Disclosure	×	×	×	
Availability	Denial of Service	×	×	×	
Authorization	Elevation of Privilege			×	

To identify which threats are applicable to a specific system, threat tree patterns can be used. For each valid intersection in Table 1, a threat tree pattern suggests the possible security-related preconditions for the STRIDE category, in order to help analysts determine the relevance of a threat for the system. An example threat tree is presented in Figure 2. Each path of the threat tree indicates a valid attack path. Note that some trees cascade. For example, the tree in Figure 2 shows the conditions that could lead to tampering threats against a process. The node indicated as a circle (or oval) in the threat tree means a root threat. These are the main STRIDE threats which, indirectly, can lead to another root threat, e.g. someone can indirectly tamper with a process by spoofing an external entity. The node indicated as a rectangle suggest a concrete threat in an attack path. The arrows connecting the nodes in general refer to a *OR* relation among the various preconditions, unless it is indicated explicitly with "AND" to refer to a *AND* relation.

Afterwards, the identified privacy threats need to be documented as misuse cases, i.e., as a collection of threat scenarios in the system.

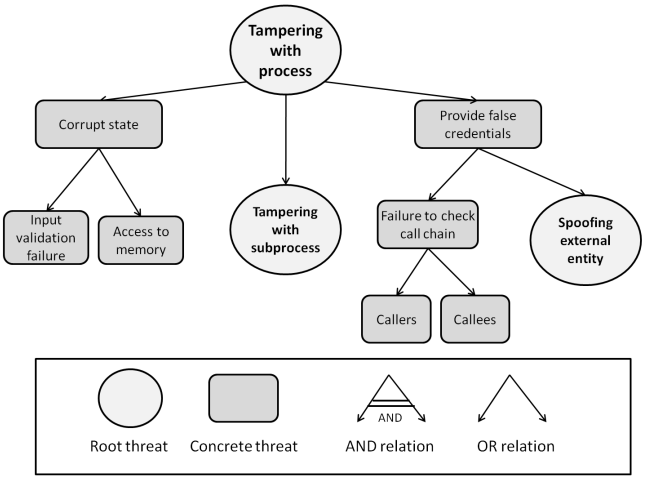

Fig. 2 Example security threat tree pattern of tampering a process [3]

Step 8: Determine risk. For each threat, the appropriate security risk level has to be determined, which can be used to define the priorities of the threats to be resolved.

Step 9: Plan mitigation. In the final step of the methodology, the risk of the threat is reduced or eliminated by introducing proper countermeasures and defenses. Mitigating a risk to the threat corresponds to eliminating one attack path in the threat tree. An overview of some possible mitigation technologies linked to each security property is provided.

These steps are security-related and should be enhanced by the corresponding privacy perspective in order to perform a privacy threat analysis. In particular, privacy assumptions need to be specified in step 3 and external privacy notes are considered in step 4. This paper proposes privacy-specific extensions to the key steps: determining privacy threat types (step 6) in Section 5.1 and identifying privacy threats (step 7) in Sections 5.2 to 7. The mitigation of privacy threats via privacy enhancing solutions (step 9) is discussed in Section 8.

2.1 Security Threat Modeling Techniques

STRIDE comes with the main advantage of an extensive, reusable knowledge base (i.e. the threat tree patterns). However, some alternatives to elicit security threats exist.

Attack trees are similar to fault trees [5]. The root node describes the high-level attack which is further decomposed in lower-level attack branches. Each node represents a step that must be successfully executed in order to complete the attack represented by the parent node. Nodes can be composed in conjunctions and disjunctions. Attack trees can have both a graphical and a textual representation.

Misuse cases [6] or abuse cases are similar to regular use cases, however with a focus on the attacker's actions. Misuse cases have a textual representation, similar to use cases, and can also be represented in a misuse case diagram, which summarizes all existing misuse cases for a certain system and their impact on the system's use cases. Both techniques can be used to elicit security threats; these techniques however do not provide methodological guidance to discover additional threats. Opdahl and Sindre [7] have made an experimental comparison between attack trees and misuse cases of which the main finding was that attack trees are more effective for finding threats. Attack trees encourage the use of standard textbook threats and decomposition in lower-level threats. Misuse case analysis focuses more on user-level and organizational threats.

KAOS [1], a goal-oriented requirements analysis framework, has been extended with anti-goals to support the modeling of threats. Such anti-goals express the goals of an attacker who tries to abuse the system. Although no actual methodology exists to determine the threats, the root anti-goals are created by negating all the positive system goals.

Next, the anti-goals are refined into trees. The formal nature of KAOS is an advantage which makes it possible to determine completeness of the (anti-)goals.

3 Our approach – the LINDDUN methodology

In this work, we propose a systemic approach for privacy threat modeling – the LINDDUN methodology – to elicit the privacy requirements of software-intensive systems and select privacy enhancing technologies accordingly. Each letter of "LINDDUN" stands for a privacy threat type obtained by negating a privacy property. Privacy properties and threats types are briefly described in Sections 4 and 5, respectively.

Figure 3 depicts the building blocks of LINDDUN. In the figure, a distinction is marked between the proposed methodology and the supporting knowledge provided to assist each step. First of all, a data flow diagram is created based on the high-level system description. This is followed by mapping privacy threats to the DFD elements using Table 4 as a guide to determine the corresponding threats. In particular, a number of privacy tree patterns from Section 6 will be proposed to detail the privacy threat instances in a designated system, by providing an overview of the most common preconditions of each threat. Next, the identified privacy threats that are relevant to the designated system are documented as misuse cases (cf. Section 7). A misuse case presents a collection of threat scenarios in the system.

The identified privacy threats that needs to be evaluated and prioritized via risk assessment. Indeed, due to both time and budget constraints, not all threats are worthy further treatment. Note that details on the risk-analysis process are beyond the scope of this work.

The last two steps comprise so-called "white hat" activities. The privacy requirements of the system are elicited from the misuse cases following the mapping in Table 6. Finally, appropriate privacy enhancing solutions are selected according to the privacy requirements. Table 7 provides an overview of the state-of-art privacy enhancing techniques and the mapping to their corresponding privacy objectives.

The fact that the LINDDUN framework and STRIDE are based on similar approaches creates synergy. Therefore, the privacy and security analysis can be closely integrated into the SDL. Nevertheless, the aforementioned LINDDUN framework for privacy can be performed independently.

4 Privacy properties

It is not the intention to propose a new taxonomy of privacy definitions in this paper. However, it is crucial to have the right basis for the proposed LINDDUN framework, therefore definitions of privacy properties are elaborately studied and reviewed in this section. The literature is rich of studies

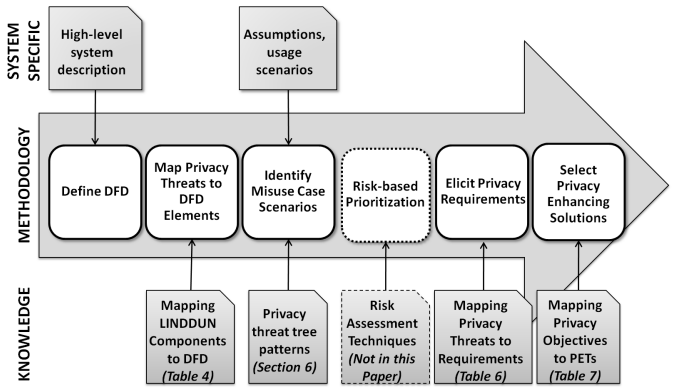

Fig. 3 The LINDDUN methodology and the required system-specific knowledge

to conceptualize privacy, and we refer interested readers to the work by Solove [8,9] for a comprehensive understanding of privacy. Most privacy properties in the LINDDUN framework comply with the terminology proposed by Pfitzmann et al. [10], as is widely recognized in the privacy research community.

4.1 Understanding privacy: hard privacy vs. soft privacy

As an abstract and subjective concept, the definition of privacy varies depending on social and cultural issues, study disciplines, stakeholder interests, and application context. Popular privacy definitions include "the right to be let alone", focusing on freedom from intrusion, and "the right to informational self-determination", allowing individuals to "control, edit, manage, and delete information about themselves and decide when, how and to what extent that information is communicated to others" [11].

Privacy can be distinguished as *hard privacy* and *soft privacy*, as proposed by Danezis [12]. The data protection goal of hard privacy refers to *data minimization*, based on the assumption that personal data is not divulged to third parties. The system model of hard privacy is that a data subject (as a security user) provides as little data as possible and tries to reduce the need to "trust" other entities. The threat model includes service provider, data holder, and adversarial environment, where strategic adversaries with certain resources are motivated to breach privacy, similar to security systems. Soft privacy, on the contrary, is based on the assumption that data subject lost control of personal data and has to trust the honesty and competence of data controllers. The data protection goal of soft privacy is to provide data security and process data with specific purpose and consent, by means of policies, access control, and audit. The system model is that the data subject provides personal data and the data controller (as a security user) is responsible for the data

protection. Consequently, a weaker threat model applies, including different parties with inequality of power, such as external parties, honest insiders who make errors, and corrupt insiders within honest data holders. An overview of hard and soft privacy solutions will be given in Section 8.

Besides conceptualizing privacy, another research challenge is to define privacy properties in software based systems. Some classical security properties are desired for building in privacy, including *confidentiality* (ensuring that information is accessible only by authorized parties), *integrity* (safeguarding the accuracy and completeness of information and processing methods), *availability* (or *censorship resistance*, ensuring information is accessible to authorized users), and *non repudiation* (ensuring one not be able to deny what one has done). The definitions of these properties can be found in ISO 17799 [13].

In addition, a number of properties are also appreciated, including *anonymity* (hiding links between identity and action or a piece of information), *unlinkability* (hiding link between two or more actions, identities and pieces of information), *undetectability* (or covertness) and *unobservability* (hiding user's activity), *plausible deniability* (opposite as non-repudiation, no others can prove one has said or done something), and *forward security* (also referred as forward secrecy and freedom from compulsion, meaning that once the communication is securely over, it cannot be decrypted any more).

We decided to include the following privacy properties in the proposed framework, namely unlinkability, anonymity and pseudonymity, plausible deniability, undetectability and unobservability, and confidentiality (hiding data content, including access control) as hard privacy properties; *user content awareness* (including feedback for user privacy awareness, data update and expire) together with *policy and consent compliance* as soft privacy properties. These properties are described in the following sections. Note that properties such as integrity, availability, and forward security are also important for privacy. However, we consider them as typical security properties; hence they are to be considered in the security engineering framework, such as STRIDE.

4.2 Unlinkability

The unlinkability property refers to hiding the link between two or more actions, identities, and pieces of information. Examples of unlinkability include hiding links between two anonymous messages sent by the same person, two web page visits by the same user, entries in two databases related to the same person, or two people related by a friendship link in a social network.

Unlinkability is defined Pfitzmann et al. as [10]: *"Unlinkability of two or more items of interest (IOIs, e.g., subjects, messages, actions, ...) from an attackers perspective means that within the system (comprising these and possibly other items), the attacker cannot sufficiently distinguish whether these IOIs are related or not."* Although it is not explicitly mentioned, the definition of unlinkability implies that the two or more IOIs are of the comparable types, otherwise it is infeasible to make the comparison.

4.3 Anonymity

Essentially, the anonymity property refers to hiding the link between an identity and an action or a piece of information. Examples are anonymous sender of an email, writer of a text, person accessing a service, person to whom an entry in a database relates, and so on.

Anonymity is defined as [10]: *"Anonymity of a subject from an attackers perspective means that the attacker cannot sufficiently identify the subject within a set of subjects, the anonymity set."* Anonymity can also be described in terms of unlinkability. If one considers sending and receiving of messages as attributes; the items of interest (IOIs) are who has sent or received which message. Then, *"anonymity of a subject with respect to an attribute may be defined as unlinkability of this subject and this attribute."* For instance, *sender anonymity* of a subject means that to this potentially sending subject, each message is unlinkable.

4.4 Pseudonymity

The pseudonymity property suggests that it is possible to build a reputation on a pseudonym and possible to use multiple pseudonyms for different purposes. Examples include a person publishes comments on social network sites under different pseudonyms and a person uses a pseudonym to subscribe to a service.

Pfitzmann et al. [10] defines pseudonymity as: *"A pseudonym is an identifier of a subject other than one of the subjects real names. Pseudonymity is the use of pseudonyms as identifiers. A subject is pseudonymous if a pseudonym is used as identifier instead of one of its real names."* Pseudonymity can also be perceived with respect to linkability. Whereas anonymity and identifiability (or accountability) are the extremes with respect to linkability to subjects, pseudonymity is the entire field between and including these extremes. Thus, pseudonymity comprises all degrees of linkability to a subject.

4.5 Plausible deniability

For privacy, plausible deniability refers to the ability to deny having performed an action that other parties can neither

confirm nor contradict. Plausible deniability from an attackers perspective means that an attacker cannot prove a user knows, has done or has said something. Sometimes, depending on the application, plausible deniability is desirable over non-repudiation, for instance, in an application used by whistleblowers, users will want to deny ever sent a certain message to protect their safety. Other examples include off-the-record conversations, possibility to deny the existence of an encrypted file, deny that a file is transmitted from a data source, or deny that a database record belongs to a person.

The relation between non-repudiation and plausible deniability is according to Roe in [14]: *"The goal of the non-repudiation service is to provide irrefutable evidence concerning the occurrence or non-occurrence of an event or action. If we believe that there is a need for this as a security service[...] we must also concede that some participants desire the opposite effect: that there be no irrefutable evidence concerning a disputed event or action."* This "complementary service" is plausible deniability. In particular, it ensures that "an instance of communication between computer systems leaves behind no unequivocal evidence of its having taken place. Features of communications protocols that were seen as defects from the standpoint of non-repudiation can be seen as benefits from the standpoint of this converse problem, which is called plausible deniability."

4.6 Undetectability and unobservability

The undetectability and unobservability properties refer to hiding the user's activities. Practical examples include, it is impossible to know whether an entry in a database corresponds to a real person, or to distinguish whether someone or no one is in a given location.

Undetectability is defined as [10]: "*Undetectability of an item of interest (IOI) from an attackers perspective means that the attacker cannot sufficiently distinguish whether it exists or not. If we consider messages as IOIs, this means that messages are not sufficiently discernible from, e.g., random noise.*" For anonymity and unlinkability, not the IOI, but only its relationship to the subject or other IOIs is protected. For undetectability, the IOIs are protected as such.

Undetectability by uninvolved subjects together with anonymity even if IOIs can be detected is defined as unobservability [10]: "*Unobservability of an item of interest (IOI) means undetectability of the IOI against all subjects uninvolved in it and anonymity of the subject(s) involved in the IOI even against the other subject(s) involved in that IOI.*" The definition suggests that unobservability is undetectability by uninvolved subjects AND anonymity even if IOIs can be detected. Consequently, unobservability implies anonymity, and unobservability implies undetectability. It means, with respect to the same attacker, unobservability reveals always only a subset of the information anonymity reveals. Later sections of this paper will focus on undetectability, since unobservability is in fact a combination of undetectability and anonymity.

4.7 Confidentiality

The confidentiality property refers to hiding the data content or controlled release of data content. Examples include transferring encrypted email, applying access control to a classified document or a database containing sensitive information.

NIST[15] describes confidentiality as following: *Confidentiality means preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information*. Although confidentiality is a security property, as the definition above states, it is also important for preserving privacy properties, such as anonymity and unlinkability. Therefore, confidentiality is also considered an important privacy property.

4.8 Content awareness

Unlike the aforesaid classical privacy properties, to our knowledge, the following two properties, namely content awareness, and policy and consent compliance, are not explicitly defined in the literature. However, we consider them important privacy objectives, due to their significance to privacy and data protection. With the emerging of Web 2.0 technologies, users tend to provide excessive information to service providers and lose control of their personal information. Therefore, the content awareness property is proposed to make sure that users are aware of their personal data and that only the minimum necessary information should be sought and used to allow for the performance of the function to which it relates.

The more personal identifiable information a data subject discloses, the higher the risk is for privacy violation. To ensure content awareness, a number of technical enforcement tools have been developed. For instance, the concept of personal information feedback tools has been promoted [16, 17] to help users gain privacy awareness and self-determine which personal data to disclose.

The Platform for Privacy Preferences Project (P3P) [18] has been designed to allow websites (as data controllers) to declare their intended use of the information that they collected about the browsing users (as data subjects). P3P addresses the content awareness property by making users aware of how personal data are processed by the data controller.

Although not necessarily privacy-oriented, another responsibility of the user, within the realm of content awareness objective, is to keep user's data up-to-date to prevent

wrong decisions based on incorrect data. This means that the data subject or the data controller (depends on applications) is responsible for deleting and updating inaccurate information. For example, it is crucial to maintain patient's data in e-health applications. Imagine a doctor forgetting to mention that the patient is a diabetic, the absence of information could cause fatal consequences for patients taking medication without considering negative side effects on diabetics.

To summarize, the content awareness property focuses on the user's consciousness regarding his own data. The user needs to be aware of the consequences of sharing information. These consequences can refer to the user's privacy, which can be violated by sharing too much personal identifiable information, as well as to undesirable results by providing incomplete or incorrect information.

4.9 Policy and consent compliance

Unlike the content awareness property focused on the user, the policy and consent compliance property requires the whole system – including data flows, data stores, and processes – as data controller to inform the data subject about the system's privacy policy, or allow the data subject to specify consents in compliance with legislation, before users accessing the system. According to the definitions from the EU Directive 95/46/EC [19]: *"Controller shall mean the natural or legal person, public authority, agency or any other body which alone or jointly with others determines the purposes and means of the processing of personal data." "The data subject's consent shall mean any freely given specific and informed indication of his wishes by which the data subject signifies his agreement to personal data relating to him being processed."*

A policy specifies one or more rules with respect to data protection. These are general rules determined by the stakeholders of the system. Consents specify one or more data protection rules as well, however, these rules are determined by the user and only relate to the data regarding this specific user. The policy and consent compliance property essentially ensures that the system's policy and the user's consent, specified in textual form, are indeed implemented and enforced.

This property is closely related to legislation. There are a number of legal frameworks addressing the raised concerns of data protection, such as the Health issued the Insurance Portability and Accountability Act (HIPAA) [20] in the United States, the Data Protection Directive 95/46/EC [19] in Europe, the Personal Information Protection and Electronic Documents Act and Privacy Act [21] in Canada, the Commonwealth Privacy Act 1988 and Privacy Amendment (Private Sector) Act 2000 [22] in Australia, and the OECD Guidelines on the Protection of Privacy and Transborder Flows of Personal Data [23].

One example of consent compliance is in e-health, for some countries, healthcare professionals are not allowed to intervene until the data subject has given informed consent for medical treatment.

There are initiatives to protect data subjects and create openness; however it is evidently important to ensure that internal rules actually comply with that promised in policies and consents. Unfortunately, few technical solutions exist to guarantee the compliance. A possible non-technical solution is to use employee contracts to enforce penalties (e.g., get fired or pay fines) to ensure compliance. Another solution is to hire an auditor to check policies compliance. Eventually, necessary legal actions can be taken by data subjects in case of noncompliance.

Breaux et al. [24] pointed out that to ensure a product that complies with its privacy and security goals, legal requirements need to be identified and refined into product requirements, and the product requirements need to be integrated into the ongoing product design and testing processes. They presented an industry case study in which requirements of Cisco products were specified to comply with Section 508 of the U.S. Workforce Investment Act (WIA) of 1998 [25]. They developed a set of qualitative metrics to rationalize the comparison of two requirements. These metrics demonstrate that alignments between legal and product requirements can be described in detail by using the goal-oriented concept of refinement. Their analysis revealed that a frame-based requirements analysis method [26], which itemizes requirements and preserves legal language, is useful to incorporate legal requirements into a manufacturer's compliance framework.

5 Mapping privacy threats to DFD

In this section, we present the privacy threat categories based on the above-mentioned privacy properties. We also discuss how to map these categories to the DFD elements.

5.1 Privacy threat categories

As shown in Table 2, the methodology considers seven types of threats. LINDDUN is the mnemonic acronym that we use.

The following section describes LINDDUN components:

1. *Linkability* of two or more items of interest (IOIs, e.g., subjects, messages, actions, etc.) allows an attacker to sufficiently distinguish whether these IOIs are related or not within the system.
2. *Identifiability* of a subject means that the attacker can sufficiently identify the subject associated to an IOI, for instance, the sender of a message. Usually, identifiability refers to a set of potential subjects, called the identifiability set [10]. In essence, identifiability is a special case

Table 2 In the LINDDUN methodology, privacy properties and the corresponding privacy threat are categorized as hard privacy and soft privacy

	Privacy properties	Privacy threats
HARD	Unlinkability	**L**inkability
	Anonymity & Pseudonymity	**I**dentifiability
	Plausible deniability	**N**on-repudiation
	Undetectability & Unobservability	**D**etectability
	Confidentiality	**D**isclosure of information
SOFT	Content awareness	content **U**nawareness
	Policy and consent compliance	policy and consent **N**oncompliance

of linkability when a subject and its attributes are involved. Identifiability is a threat to both anonymity and pseudonymity.

3. *Non-repudiation*, in contrast to security, this is a threat for privacy. Non-repudiation allows an attacker to gather evidence to counter the claims of the repudiating party, and to prove that a user knows, has done or has said something.
4. *Detectability* of an IOI means that the attacker can sufficiently distinguish whether such an item exists or not. If we consider messages as IOIs, it means that messages are sufficiently discernible from random noise.
5. *Information Disclosure* threats expose personal information to individuals who are not suppose to have access to it.
6. *Content Unawareness* indicates that a user is unaware of the information disclosed to the system. The user either provides too much information which allows an attacker to easily retrieve the user's identity or inaccurate information which can cause wrong decisions or actions.
7. *Policy and consent Noncompliance* means that even though the system shows its privacy policies to its users, there is no guarantee that the system actually complies to the advertised policies. Therefore, the user's personal data might still be revealed.

5.2 Mapping privacy threat categories to the system

This section provides the guidelines to identify privacy threats of a software based system. First, a Data Flow Diagram (DFD) is created in correspondence to the application's use case scenarios. Second, privacy threats are mapped to the DFD.

5.2.1 Creating Application DFD Based On Use Case Scenarios

DFD is chosen to represent a software system based on two reasons. First, DFD is proven to be sufficiently expressive in a number of case studies examined by the authors. Second, DFD is also used by the SDL threat modeling process, hence by deploying the same modeling technique an interesting synergy can be created between the proposed framework and the SDL process.

Table 3 DFD elements in the Social Network 2.0 application

Entity	User
Process	Portal Social network service
Data Store	Social network DB
Data Flow	User data stream (user-portal) Service data stream(portal-service) DB data stream (service DB)

Table 4 Mapping LINDDUN components (privacy threats) to DFD element types (E-Entity, DF-Data flow, DS-Data store, P-Process)

Threat categories	E	DF	DS	P
Linkability	×	×	×	×
Identifiability	×	×	×	×
Non-repudiation		×	×	×
Detectability		×	×	×
Information **D**isclosure		×	×	×
content **U**nawareness	×			
policy/consent **N**oncompliance		×	×	×

Running example: Social Network 2.0

In our running example Social Network 2.0, Alice is a registered user of a social network. Each time Alice updates her friends list, she first connects to the social network's web portal. Accordingly, the portal communicates with the social network's server, and eventually, the friendship information of Alice and all other users of that social network is stored in a database.

The DFD for the Social Network 2.0 application was already presented in Figure 1 of Section 2. Table 3 lists the DFD elements.

The creation of the DFD is an important part in the analysis. If the DFD was incorrect, the analysis results would be wrong as well. Since privacy focuses on the protection of user's personal information, it is important to consider where the information will be stored or passed by, as these are the crucial elements for building in privacy.

5.2.2 Mapping Privacy Threats to DFD

After the DFD elements are listed, we identify the privacy threat categories for each DFD element by following the mapping depicted in Table 4. Each intersection marked with the symbol × indicates a potential privacy threat at a corresponding DFD element in the system.

In essence, each DFD element is subject to certain privacy threats, and the nature of the potential privacy threat is determined by the DFD element type. For example, a data flow is subject to a number of privacy threats such as identifiability, linkability, detectability, non-repudiation, and information disclosure. The following sections will explain how privacy threats affect DFD elements. More threat scenarios corresponding to our running example will be discussed in Section 7.

The nature of *linkability* indicates that the threat affects DFD elements by pair. In other words, linkability of a DFD element refers to a pair (x_1, x_2), where $x \in \{E, DF, DS, P\}$ is the linkable IOI. Obviously, linkability at entity, from an attackers perspective means that within the system (comprising these and possibly other items), the attacker can sufficiently distinguish whether these entities are related or not. Similar description applies for that of data flow, data store, and process.

The *identifiability* threat affects all four DFD elements, such that each DFD element is made explicit as the attributes that identifiability (or its opposite property anonymity) relates to, by forming a pair with a subject. Essentially, identifiability at each DFD element refers to a pair (x, y), where $x \in \{E\}$ is the identifiable subject, and $y \in \{E, DS, DF, P\}$ is the attribute identifiability relates to. For example, identifiability at entity refers to a pair (E, E), meaning to identify an entity within a set of entities. Identifiability at data flow refers to a pair (E, DF), meaning that a message is linkable to a potentially sending or receiving subject. Identifiability at data store refers to a pair (E, DS), meaning that a database entry is linkable to a potential data holder or subject. Identifiability at process refers to a pair (E, P), meaning that a process is linkable to a potentially accessing subject.

Non-repudiation, opposite of plausible deniability, is a privacy threat that affects the DFD elements of data flow, data store and process. Non-repudiation might be appreciated for some system but undesirable for others. It depends on the system requirements. For e-commerce applications, non-repudiation is an important security property. Imagine a situation where a buyer signs for a purchased item upon receipt, the vendor can later use the signed receipt as evidence that the user received the item. For other applications, such as off-the-record conversations, participants may desire plausible deniability for privacy protection such that there will be no record to demonstrate the communication event, the participants and the content. In this scenario, non-repudiation is a privacy threat. Even though entity is the only DFD element being able to (non-)repudiate, the non-repudiation privacy threat actually occurs at data flow, data store, and process. Similar to linkability and identifiability, non-repudiation at each DFD element refers to a pair (x, y), where $x \in \{E\}$ is the non-repudiating subject, and $y \in \{DS, DF, P\}$ is the attribute it relates to.

Table 5 Determining privacy threats for DFD elements within the Social Network 2.0 application (From left to right: L-Linkability, I-Identifiability, N-Non Repudiation, D-Detectability, D-Information Disclosure, U-Content Unawareness, N-Consent/policy Noncompliance)

	Threat target	L	I	N	D	D	U	N
Data Store	Social network DB	1	4	×	×	7		10
Data Flow	User data stream (user – portal)	2	5	×	×	8		10*
	Service data stream (portal – service)	×	×	×	×	×		10*
	DB data stream (service – DB)	×	×	×	×	×		10*
Process	Portal	×	×	×	×	×		10*
	Social network service	×	×	×	×	×		10*
Entity	User	3	6				9	

Detectability threats occur at data flow, data store, and process, meaning that the attacker can sufficiently distinguish whether it exists or not. Though in some applications, techniques such as covert channel and steganography can be used to protect both messages (data flow) and communicating parties (entity), in this case the threat actually occurs at data flow instead of entity. In other words, the asset we want to protect against the detectability threat includes data flow, data store, and process.

Information disclosure threats affect data flow, data store, and process, referring to the exposure of information at these DFD elements to individuals who are not supposed to have access to it.

The *content unawareness* threat is related to entity, since the entity (data subject or data controller) is actually responsible to provide the necessary consents to process personal data and update or delete the expired information.

Policy and consent noncompliance is a threat that affects system as a whole, because each system component (including data flow, data store and process) is responsible to ensure that actions are taken in compliance with privacy policies and data subject's consents.

Running example: Social Network 2.0

Considering the Social Network 2.0 application, the list of generic privacy threats to the modeled system is depicted in Table 5. This is obtained by gathering the elements from Table 3 and then determining the susceptible threats with Table 4.

The intersections marked with × in Table 5 are potential threats that have been considered as irrelevant to the specific usage scenario. Each intersection that is indicated with a number (1 to 10) in Table 5 shows that there will be a privacy threat at the corresponding DFD element. These items marked with a number are the threats which we will actually

consider. The number represents the ID of the generic threat and will be used later for ease of reference.

Primarily, we assume that DFD elements within the trust boundary (marked as dashed line in Figure 1) are trustworthy. We trust the processes within the boundary, as well as all data flows in the trust boundary. Therefore, we will not discuss linkability, identifiability, and information disclosure threats on these elements. We however do not trust the user and its communication with the portal and we also want to protect the data store containing all the user's information.

Moreover, non-repudiation and detectability threats are considered irrelevant for social networks. Presumably, it depends on what privacy properties are required for a particular social network system. In case plausible deniability and undetectability would be desirable for a certain application, we should still consider these threats for each DFD element accordingly.

Following the above reasoning, ten threats will be examined in detail in Section 7, and they are numbered in Table 5. Note that some items are indicated with a $10*$. This means that the policy and consent noncompliance threat affects the system as a whole (including data flow, data store and process). However, we will only illustrate one misuse case for this threat in this paper.

6 Detailing privacy threats via threat tree patterns

This section presents an extensive catalog of threat tree patterns that can be used to detail the privacy threats to a realistic system. For each marked intersection in Table 4, a threat tree pattern exists showing the detailed preconditions for this specific threat category to materialize. The preconditions are hence vulnerabilities that can be exploited for a privacy attack scenario.

The present catalog is based on the state-of-art privacy developments and the threat trees reflect common attack patterns and help application designers think about privacy conditions in the system. However, the threat trees depicted in this section present the best effort so far. The catalog is subject to continuous improvement in order to reflect newly discovered threats. Further, the catalog is meant to be updated as new results are available from the industrial validation experiments.

6.1 Linkability of entity

Linkability of entity refers to an attacker can sufficiently distinguish whether two or more entities are related or not within the system. This implies that different pseudonyms can be linked to each other. The threat tree pattern is depicted in Figure 4. One precondition is that data flow or data store is not fully protected (e.g. unencrypted), which leads to the

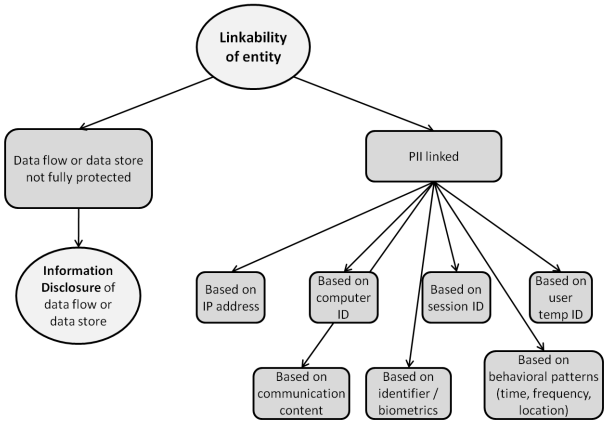

Fig. 4 Threat tree for linkability of an entity

Information Disclosure threat of data flow and data store. The second precondition is that Personal Identifiable Information (PII) can be linked, e.g. based on user temporary ID, IP address, behavioral patterns such as time, frequency and location, session ID, identifier and biometrics, computer ID, communication content or any combination of these factors. The aforementioned data store refers to the identity management system's database or any other database which contains personal identifiers of users. Having accessed such a data store, the attacker could easily link different pseudonyms to the same user

6.2 Linkability of data flow

Linkability of data flow threat tree, as presented in Figure 5, suggests two preconditions. One precondition is that data flows are not fully protected (e.g. unencrypted), which leads to information disclosure of data flow; and communications are linkable due to little or insecure anonymity systems deployed. The other precondition is that communication can be linked. When no anonymous communication is deployed, basically the same preconditions apply as for the linkability of entity threat. Messages are linked to each other by user's identifiable information (e.g. based on user temporary ID, IP address, behavioral patterns such as time, frequency and location, session ID, identifier and biometrics, computer ID, communication content or any combination of these factors). Alternatively, when an insecure anonymity system is deployed, traffic analysis is possible to extract information out of patterns of traffic; passive attacks (e.g. long-term intersection attacks, traffic correlation and confirmation, fingerprinting, epistemic attacks (route selection), and predecessor attacks) and active attacks (e.g. N-1 attacks, Sybil attack, traffic watermarking, tagging attack, replay, and DoS attack) are possible to link entities together. An overview of these attacks can be found in [27].

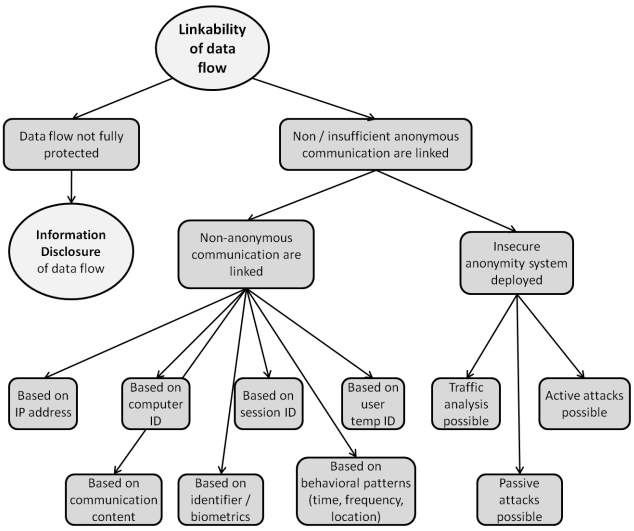

Fig. 5 Threat tree for linkability of a data flow

6.3 Linkability of data store

Two preconditions correspond to the threat of *linkability of a data store*, as shown in Figure 6. First, there is insufficient access control of the data store leading to the information disclosure threat at a data store. Second, insufficient data anonymization is applied or strong data mining is possible in the data store, meaning that the stored information still contains sufficient references to the corresponding data subject, which makes it possible to link different data items to each other within the same database. Another possibility is that data can be linked from one database to another, and hence re-identification [28] is possible.

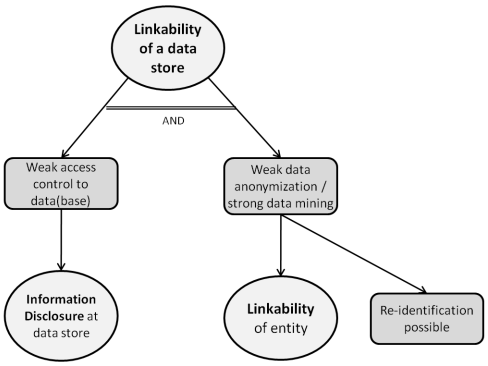

Fig. 6 Threat tree for linkability of a data store

6.4 Linkability of process

The threat tree of *linkability of process* suggests that the only way to prevent different actions being linked to the same subject is by gaining access to the process, as illustrated in Figure 7.

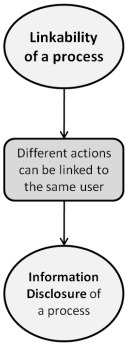

Fig. 7 Threat tree for linkability of a process

6.5 Identifiability of entity

Figure 8 shows the threat tree pattern for *identifiability of entity*. It gives an overview of the most common situations where an identifiability threat can occur at an entity. A first precondition is when the e-id is used as login (i.e., the user's actual identity is used), meanwhile the data flow between the user and login portal is not sufficiently protected (i.e., the threat of information disclosure of data flow), and thus the user's identity will be exposed. A second possibility occurs when a secret (e.g. a password) is used as log-in and the relationship between this secret and the user can be revealed. This can happen if the identity management database in not secure (e.g. passwords are stored in clear); if the communication channel is insecure and the communicated passwords are weak and can be connected to the user (e.g. using a birthdate as password); or if replay attacks are possible (e.g. a keylogger is installed, the communication can be eavesdropped or the person entering the secret can be observed). A third possible precondition for the threat is the use of a token as log-in which is weakly implemented or physically insecure. The final precondition is that biometrics is used as log-in, which means that biometrics is retrievable and can be linked to an entity. It is due to information disclosure of identity management database or data flow which contains biometrics, and linkability at data store.

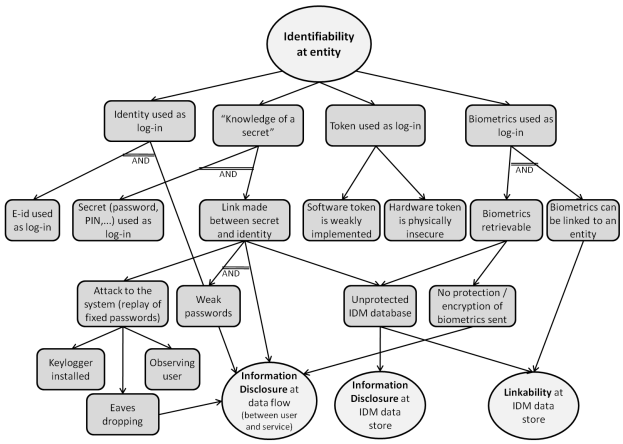

Fig. 8 Threat tree for identifiability of an entity

6.6 Identifiability of data flow

The threat tree of *identifiability of data flow* is presented in Figure 9. Similar to the linkability of data flow threat tree, identifiability of data flow is possible when data flows are not fully protected, which leads to information disclosure of data flow; or when the communication can be traced to an entity due to little or insecure anonymity system deployed. When no anonymity system is deployed, communication can be traced to an entity by means of identifiable information (e.g. based on user temporary ID, IP address, behavioral patterns such as time, frequency and location, session ID, identifier and biometrics, computer ID, communication content or any combination of these factors). Alternatively, when an insecure anonymity system is deployed, traffic analysis, passive attacks, and active attacks [27] are possible to identify the particularly entity of interests.

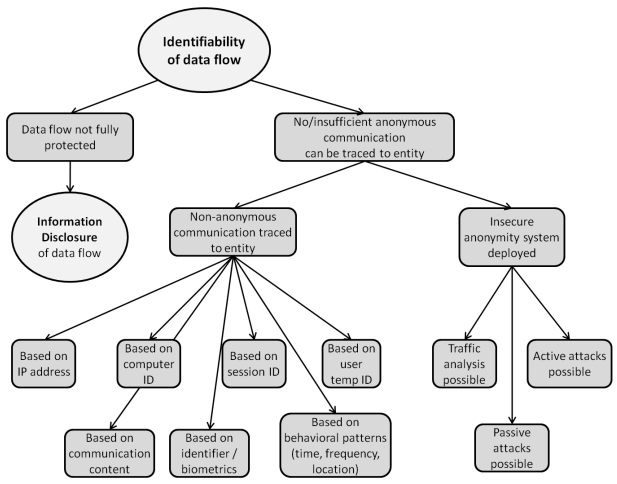

Fig. 9 Threat tree for identifiability of a data flow

6.7 Identifiability of data store

The preconditions for *identifiability of data store*, as presented in Figure 10, are similar to the preconditions of linkability at a data store. Either there is insufficient access control of the data store which refers to the information disclosure threat of a data store; or insufficient data anonymization or strong data mining techniques where applied in the data store, which means that the information stored still contains sufficient references to the corresponding person to reveal the person's identity, and hence it refers to the identifiability threat of entity or the data can be linked with other relevant information which can lead to re-identification of the data subject.

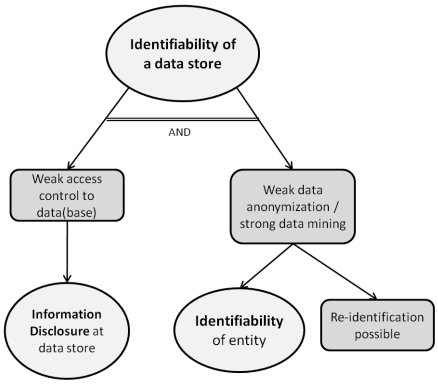

Fig. 10 Threat tree for identifiability of a data store

6.8 Identifiability of process

The threat tree of *identifiability of process* is presented in Figure 11. The only condition for the identifiability threat at a process is when access to the process is not sufficiently secured, and thus it refers to the threat of information disclosure of a process.

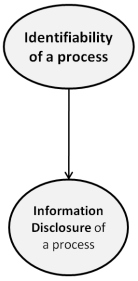

Fig. 11 Threat tree for identifiability of a process

6.9 Non-repudiation of data flow

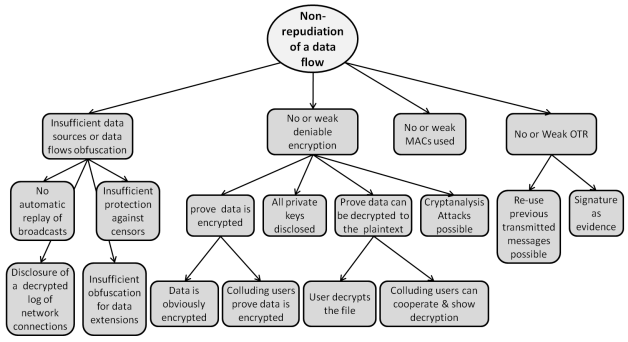

Fig. 12 Threat tree for Non-repudiation of a data flow

Four general preconditions can be applied to the threat tree of *non-repudiation of data flow*, as presented in Figure 12. One condition is insufficient obfuscation for data sources or data flows, which means that the attacker can gain access to at least part of the data flow or data source. This can occur in a number of cases, for example, there is no automatic replay of broadcasts, such that the sender of a file is sufficiently distinguishable from those who are merely relaying it. Another example is when a complete decrypted log of all network connections to and from a user's computer is disclosed, resulting in the disclosure of the origin of data flow. The final examples are that there is insufficient protection against censors or insufficient obfuscation of data extensions, such that operators or users of the network are able to know where the data comes from.

The second precondition of this threat is that little or a weak deniable encryption technique is used to protect data flow. One possible attack path is to prove data is encrypted, either due to the encrypter proves the data is obviously an encryption or colluding users prove together that the data is encrypted. The second attack path is to prove data can be decrypted to a valid plain text, which can occur when the encrypter decrypts the file or colluding users can cooperate and show the decrypted message. The third attack path shows that all private keys are disclosed, and the last path suggests that cryptanalysis is possible to attack the used encryption scheme.

The third condition is that there are little or weak message authentication codes (MAC) used to ensure integrity of data flow content, such that an attacker can forge authentic looking messages and pretend that a certain data flow comes from a subject.

The final precondition indicates that there is little or a weak Off-the-Record Messaging (OTR) used, such that in a conversation it is not possible to provide both deniability for the conversation participants and confidentiality of conversations content at the same time. Possible attack paths include replaying of previous transferred messages, and the use of signatures to demonstrate communication events, participants and communication content.

6.10 Non-repudiation of data store

The threat tree for *non-repudiation of data store* is depicted in Figure 13. Three preconditions can apply to this threat, namely a weak access control to the database, which leads to the threat of information disclosure at the data store; little or a weak deniable encrypted is used to protect the data, such that data can be proven to be an encryption or can be decrypted to a valid plaintext; and subjects with deniability are not able to edit data in the database to cover their tracks, and it can be either impossible to remove or alter the user's own data or impossible to remove or alter someone elses data concerning the user himself.

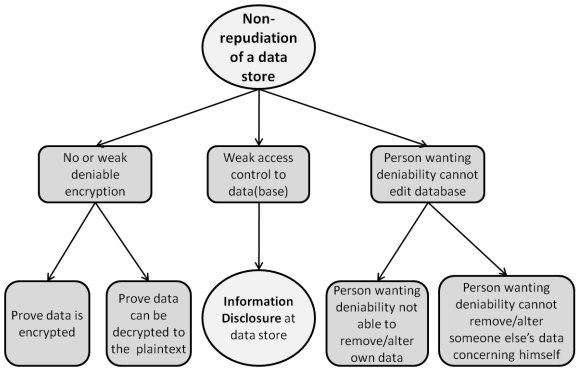

Fig. 13 Threat tree for Non-repudiation of a data store

6.11 Non-repudiation of process

Non-repudiation of process, as depicted in Figure 14 can be achieved in two ways: either the process looses its confidentiality and information disclosure attacks at the process are possible, or the process uses a secure log to create an overview of all actions, which can evidently be traced back to the user.

6.12 Detectability of data flow

The threat tree of *detectability of data flow*, as depicted in Figure 15 suggests five preconditions for the threat to occur. One condition is that the system lacks a covert channel. This can happen when the covert channel uses too much

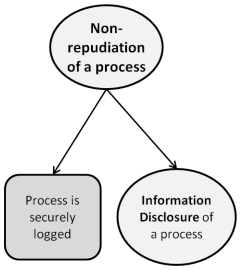

Fig. 14 Threat tree for Non-repudiation of a process

bandwidth from a legitimate channel, resulting in the detection of the covert communication. It can also be because the patterns or characteristics of the communications medium of the legitimate channel are controlled or examined by legitimate users, e.g. checking file opening and closing operations patterns or watching the timing of requests, such that covert communication is detected. The second condition is side channel analysis on timing information, power consumption, electromagnetic leaks, as an extra source of information which can be exploited to detect the communication. The third condition occurs when a weak information hiding techniques are used, which makes a number of steganalysis attacks possible. Another condition is when there is no or insufficient dummy traffic sent at some lower layer of communication network, such that messages fail to appear random for all parties except the sender and the recipient(s). Last but not least, the threat can occur because of a weak spread spectrum communication, resulting in deficiencies in the establishment of secure communications, resistance to natural interference and jamming, and detection prevention.

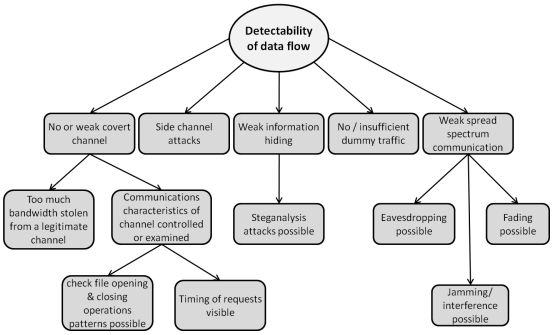

Fig. 15 Threat tree for detectability of a data flow

6.13 Detectability of data store

As shown in Figure 16, *detectability threats in a data store* can occur if there is insufficient access control, which leads to the information disclosure threats for security, or if insufficient information hiding techniques are applied, such as information from a data store is revealed due to weak steganography algorithms employed.

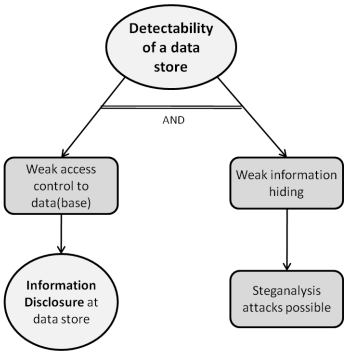

Fig. 16 Threat tree for detectability of a data store

6.14 Detectability of process

Similar to the previously described threats related to a process, the *detectability of process* threat, depicted in Figure 17, also refers to the threat of information disclosure of process.

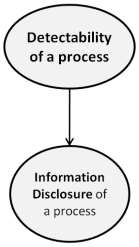

Fig. 17 Threat tree for detectability of a process

6.15 Information disclosure of data flow, data store, and process

The threat tree concerning *information disclosure of data flow, data store, and process*, depicted in Figure 18, refers to the security threat tree of information disclosure. This illustrates the fact that privacy properties are part of security properties, and privacy may depend on security. For more information about the information disclosure threats we refer to SDL [3].

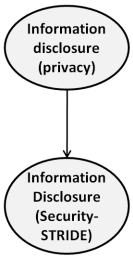

Fig. 18 Threat tree for Information Disclosure

6.16 Content unawareness of entity

Content unawareness of an entity can occur in two situations: either the date subject provides more personal identifiable information than required, which has a negative influence on all the hard privacy objectives; or data subject does not keep information updated or does not remove the outdated information, and it can lead to wrong actions when decisions are based on this information.

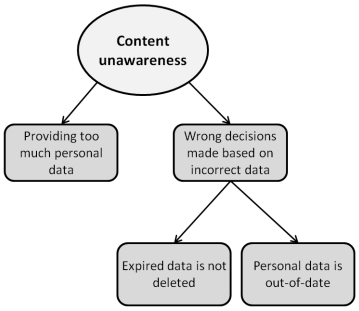

Fig. 19 Threat tree for Content Unawareness

6.17 Consent and policy noncompliance of the system (data flow, process and data store)

The user's privacy can be violated when internal system rules do not correspond to privacy policies provided to the user. This can occur when an attacker tampers with the internal policies, which is actually a security threat; or when the policy rules are incorrectly managed or updated (according to the user's requests) by the system administrator.

7 Documenting Threats Scenarios in Misuse Cases

Threat tree patterns are used to detail the generic LINDDUN threat categories into specific threat instances that can occur in a system. Furthermore, some threat instances could have been discarded during the risk-analysis step. The result of

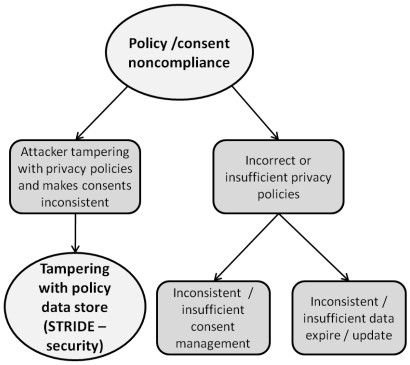

Fig. 20 Threat tree for policy and consent noncompliance

the above process should be a collection of threat scenarios that need to be documented. To this aim, misuse cases can be used. In particular, a misuse case can be considered as a use case from the misactor's point of view. A misactor is someone who intentionally or unintentionally initiates the misuse case. Alexander [29] provides some example misuse cases, together with the corresponding (positive) use cases. We chose misuse cases because they represent a well established technique to elicit requirements, and a number of support tools exist as well.

The structure of a misuse case, which is based on the template provided by Sindre and Opdahl [6] is described below:

Summary: provides a brief description of the threat.

Assets, stakeholders and threats: describes the assets being threatened, their importance to the different stakeholders, and what is the potential damage if the misuse case succeeds.

Primary misactor: describes the type of misactor performing the misuse-case. Possible types are insiders, people with a certain technical skill, and so on. Also, some misuse case could occur accidentally whereas other are most likely to be performed intentionally.

Basic Flow: discusses the normal flow of actions, resulting in a successful attack for the misactor.

Alternative Flows: describes the other ways the misuse can occur.

Trigger: describes how and when the misuse case is initiated.

Preconditions: precondition that the system must meet for the attack to be feasible.

The preconditions refer to the leaf nodes of the threat tree patterns and the basic (and alternative) flow describes how a miuser could exploit these weaknesses of the system in order to mount an attack.

Running example: Social Network 2.0

In our running example, we assume that communication and processes within the social network service provider are

trustworthy (see the trust boundary in the DFD depicted in Figure 1). However, we want to protect the data store against information disclosure. The data controllers could be users, social network providers, and application providers.

To illustrate how to create a misuse case based on the threat tree patterns, consider the threat tree of linkability at the data store (see Figure 6). The tree illustrates that in order to be susceptible to this threat, neither the data store is sufficiently protected against information disclosure nor sufficient data anonymization techniques are employed. These are the preconditions of the misuse case. To create the attack scenarios, it is clear that the attacker first needs to have access to the data store, and secondly, either the user (as the data subject) can be re-identified (as the basic flow) or the pseudonyms can be linkable (as the alternative flow). The aforementioned misuse case is presented in this section. The additional nine misuse cases applicable to the social network example are described in Appendix A.

Title: MUC 1 – Linkability of social network database (data store)
Summary: Data entries can be linked to the same person (without necessarily revealing the persons identity)
Assets, stakeholders and threats: Personal Identifiable Information (PII) of the user.

- The user:
  - Data entries can be linked to each other which might reveal the persons identity
  - The misactor can build a profile of a user's online activities (interests, actives time, comments, updates, etc.)

Primary misactor: skilled insider / skilled outsider
Basic Flow:

1. The misactor gains access to the database
2. The misactor can link the data entries together and possibly re-identify the data subject from the data content

Alternative Flow:

1. The misactor gains access to the database
2. Each data entry is linked to a pseudonym
3. The misactor can link the different pseudonyms together (linkability of entity)
4. Based on the pseudonyms, the misactor can link the different data entries

Trigger: by misactor, can always happen.
Preconditions:

- no or insufficient protection of the data store
- no or insufficient data anonymization techniques or strong data mining applied

Note that formulating soft privacy threats is less straightforward and requires some out-of-the-box thinking for suitable (non-)technical solutions. We refer the reader to misuse cases 9 and 10 in the appendix as an example of the latter case.

7.1 Risk Assessment

Similarly to STRIDE, LINDDUN can suggest a (large) number of documented threats. Before the process moves forward, the identified threats must be prioritized. Only the important ones should be considered for inclusion in the requirements specification and, consequently, in the design of the solution. Risk assessment techniques provide support for this stage. In general, risk is calculated as a function of the likelihood of the attack scenario depicted in the MUC (misuse case) and its impact. The risk value is used to sort the MUCs: the higher the risk, the more important the MUC is.

The LINDDUN framework (similarly to STRIDE) is independent from the specific risk assessment technique that is used. The analyst is free to pick the technique of choice, for instance the OWASP's Risk Rating Methodology [30], Microsoft's DREAD [31], NIST's Special Publication 800-30 [32], or SEI's OCTAVE [33]. These techniques leverage the information contained in the MUC, as the involved assets (for the impact), and the attacker profile as well as the basic/alternative flows (for the likelihood). Many of the abovementioned techniques include privacy considerations when assessing the impact of a threat. However, as a research challenge, a privacy-specific risk assessment technique is worthwhile to be investigated, as the on-field experience reveals any inadequacy of state-of-the-art techniques. This goes beyond the scope of this work.

8 From threat analysis to privacy enhancing solutions

This section explains the elicitation of privacy requirements from threat analysis and the selection of mitigation strategies and techniques based on privacy objectives.

8.1 Eliciting Privacy Requirements: From Privacy Threat Analysis to Mitigation Strategy

Misuse cases describe the relevant (risk-wise) threat scenarios for the system. The preconditions are based on the threat tree patterns and the basic and alternative flows are inspired by the system's use cases.

As a next step, the system's (positive) requirements can be extracted from the misuse cases. To this aim, the specification of the privacy requirements is facilitated by Table 6, which maps the types of threats scenarios to types of privacy requirements. Note that the table is a refinement of the more generic objectives in Table 2.

Table 6 Privacy objectives based on LINDDUN threat types (E-Entity, DF-Data Flow, DS-Data Store, P-Process)

LINDDUN threats	Elementary privacy objectives
Linkability of (E,E)	Unlinkability of (E,E)
Linkability of (DF,DF)	Unlinkability of (DF,DF)
Linkability of (DS,DS)	Unlinkability of (DS,DS)
Linkability of (P,P)	Unlinkability of (P,P)
Identifiability of (E,E)	Anonymity / pseudonymity of (E,E)
Identifiability of (E,DF)	Anonymity / pseudonymity of (E,DF)
Identifiability of (E,DS)	Anonymity / pseudonymity of (E,DS)
Identifiability of (E,P)	Anonymity / pseudonymity of (E,P)
Non-repudiation of (E,DF)	Plausible deniability of (E,DF)
Non-repudiation of (E,DS)	Plausible deniability of (E,DS)
Non-repudiation of (E,P)	Plausible deniability of (E,P)
Detectability of *DF*	Undetectability of *DF*
Detectability of *DS*	Undetectability of *DS*
Detectability of *P*	Undetectability of *P*
Information Disclosure of *DF*	Confidentiality of *DF*
Information Disclosure of *DS*	Confidentiality of *DS*
Information Disclosure of *P*	Confidentiality of *P*
Content Unawareness of *E*	Content awareness of *E*
Policy and consent Noncompliance of the system	Policy and consent compliance of the system

8.2 From Privacy Requirements to Privacy Enhancing Solutions

Similarly to security, privacy requirements can be satisfied via a range of solution strategies:

1. *Warn the user* could be a valid strategy for lower risk (but still relevant) threats. However precautions have to be taken so that users, especially nontechnical ones, do not make poor trust decisions.
2. *Removing or turning off the feature* is the only way to reduce the risk to zero. When threat models indicate that the risk is too great or the mitigation techniques are untenable, it is best not to build the feature in the first place, in order to gain a balance between user features and potential privacy risks.
3. *Countering threats with either preventive or reactive privacy enhancing technology* is the most commonly used strategy to solve specific issues.

This section mainly focuses on the last strategy. When countering threats with technology is chosen as the mitigation strategy, system designers have to identify the sound and appropriate privacy enhancing technology (PET). We summarize the state-of-art PETs in Table 7 and map these techniques to each of the corresponding privacy requirements of Table 6. As a result, improved guidance is provided to the system designers over the solution selection process.

Note that the PETs categorization is inspired by the taxonomies proposed in [34,35]. Further, Table 7 introduces some key primitives of hard privacy technologies and the state-of-art of soft privacy technologies. New privacy enhancing solutions keep emerging; therefore a complete list of PETs and best practices for choosing the appropriate mitigation is beyond the scope of this paper. The latest development of privacy enhancing technologies can be found at [36].

In summary, privacy protection solutions boil down to either technical or legal enforcement. In general, privacy technology enables functionality while offering the highest protection for privacy. Further, *Hard Privacy Technology* provides cryptographically strong protections for privacy, assumes no unnecessary leakage of information, and replies on massive distribution of trust excluding potential adversary and privacy violators. *Soft Privacy Technology* (e.g. privacy policy and feedback tools in Table 7) offers protections against mass surveillance and violations, assumes data subjects sharing of personal data is necessary, and employs a weaker adversary model.

Running example: Social Network 2.0

Table 8 summarizes the selection of PETs based on the privacy requirements elicited in our running example. It is possible that a more business oriented example would suggest different mitigation strategies. Nevertheless, we hope the example depicted in this section can illustrate how the proposed framework can be applied in real life applications.

In an attempt to make the running example more accessible to the reader, the system model, the misuse cases, and the mitigation techniques of the Social Network 2.0 are largely simplified due to the assumption that the social network providers are semi-trustworthy (i.e., the adversary model consists of external parties, data holder, honest insiders who make errors, and corrupt insiders). If different assumptions would hold, different misuse cases should be identified with a distinct mitigation approach. For instance, if we apply a smaller trust boundary and assume that the social network provider is totally untrustworthy, then extra privacy requirements and a stronger threat model would be considered. One possible misuse case would be that the malicious social network provider, as an attacker, takes advantage of profiling user's personal data for its own benefits. In that scenario, one solution could be building a security agriculture out of smart clients and an untrusted central server to removes the need for faith in network operators and gives users control of their privacy [84]. Another solution could be using encryption to enforce access control for users' personal information based on their privacy preferences [85,86].

Another research discussion is concerning practicality to build user privacy feedback tools. In short, from a technical point of view, feedback could be realized by means of data mining techniques (e.g., k-anonymity model) to coun-

Table 7 Mapping privacy objectives with privacy enhancing techniques (U – Unlinkability, A – Anonymity / Pseudonymity, P – Plausible deniability, D – Undetectability / unobservability, C – Confidentiality, W – Content Awareness, O – Policy and consent compliance of the system)

	Mitigation techniques: PETs	U	A	P	D	C	W	O
Anonymity system	Mix-networks (1981) [37], DC-networks (1985) [38,39], ISDN-mixes [40], Onion Routing (1996) [41], Crowds (1998) [42], Single proxy (90s) (Penet pseudonymous remailer (1993-1996), Anonymizer, SafeWeb), anonymous Remailer (Cipherpunk Type 0, Type 1 [43], Mixmaster Type 2 (1994) [44], Mixminion Type 3 (2003) [45]), and Low-latency communication (Freedom Network (1999-2001) [46], Java Anon Proxy (JAP) (2000) [47], Tor (2004) [48])	×	×			×		
	DC-net & MIX-net + dummy traffic, ISDN-mixes [40]	×	×		×	×		
	Broadcast systems [49,50] + dummy traffic	×	×		×			
Privacy preserving authentication	Private authentication [51,52]	×	×					
	Anonymous credentials (single show [53], multishow [54])	×	×					
	Deniable authentication [55]	×	×	×				
	Off-the-record messaging [56]	×	×	×		×		
Privacy preserving cryptographic protocols	Multi-party computation (Secure function evaluation) [57,58]	×				×		
	Anonymous buyer-seller watermarking protocol [59]	×	×			×		
Information retrieval	Private information retrieval [60] + dummy traffic	×	×		×			
	Oblivious transfer [61,62])	×	×			×		
	Privacy preserving data mining [63,64]	×	×			×		
	Searchable encryption [65], Private search [66]		×			×		
Data anonymization	K-anonymity model [28,67], l-Diversity [68]	×	×					
Information hiding	Steganography [69]	×	×		×			
	Covert communication [70]	×	×		×			
	Spread spectrum [71]	×	×		×			
Pseudonymity systems	Privacy enhancing identity management system [72]	×	×					
	User-controlled identity management system [73]	×	×					
	Privacy preserving biometrics [74]	×	×			×		
Encryption techniques	Symmetric key & public key encryption [75]					×		
	Deniable encryption			×		×		
	Homomorphic encryption [76]					×		
	Verifiable encryption [77]					×		
Access control techniques	Context-based access control [78]					×		
	Privacy-aware access control [79,80]					×		
Policy and feedback tools	Policy communication (P3P [18])							×
	Policy enforcement (XACML [81], EPAL [82])							×
	Feedback tools for user privacy awareness [16,17,83]						×	
	Data removal tools (spyware removal, browser cleaning tools, activity traces eraser, harddisk data eraser)						×	

termeasure user identification and data profiling attacks. It compares data user sends to the social network with a whole set of data composed of data from all networks users, and checks the "uniqueness" of personal identifiable information (PII) of the user. With a unique PII, a user has a higher probability to be identified. Then it warns users each time their activities provoke privacy risks, e.g. shows a risk level of identifiability by posting a message "you are about to leave the anonymity safe zone". There are some research incentives for feedback systems for social networks [16,17,83]. However, this concept implies a paradox that in order to ensure accurate feedback, the feedback tool itself should be a "perfect attacker" that knows all the data from all users. Due to the space and scope limit of this paper, we cannot discuss this in detail. We encourage interested readers to formalize the feedback system model and investigate whether it is technically realistic to realize the feedback concept and beyond which threshold a feedback could be satisfactory. Intuitively speaking, the aforementioned feedback concept is not about technical problem purely but more an education

Table 8 Social Network 2.0 example: from misuse cases to privacy requirements and suggested mitigation strategies and techniques

No.	Misuse cases	Privacy requirements	Suggested mitigation strategies and techniques
1	Linkability of social network data store	Unlinkability of data entries within the social network database	Apply data anonymization techniques, such as k-anonymity [67].
		Protection of data store	Enforce data protection by means of relationship-based access control [79]
2	Linkability of data flow of the user data stream (user-portal)	Unlinkability of messages of user-portal communication; channel confidentiality	Deploy anonymity system, such as TOR [48].
3	Linkability of entities the social network users	Unlinkability of different pseudonyms (user IDs) of social network users; channel confidentiality.	1) Technical enforcement: deploy anonymity system, such as TOR [48], for communication between user and social network web portal; 2) User privacy awareness: inform users that revealing too much information online can be privacy invasive.
4	Identifiability at the social network data store	Anonymity of social network users such that the user will not be identified from social network database entries	Protection of the data store, by applying data anonymization techniques, such as k-anonymity [67].
		Protection of data store	Enforce data protection by means of relationship-based access control [79]
5	Identifiability at data flow of user data stream (user-portal)	Anonymity of social network users such that the user will not be identified from user-portal communication by content; channel confidentiality	Deploy anonymity system, such as TOR [48], for communication between user and social network web portal.
6	Identifiability of the social network users	Pseudonymize users IDs	1) Apply secure pseudonymization techniques to issue pseudonyms as user IDs; 2) User privacy awareness: inform users using real ID has a risk for privacy violation.
		Use identity management to ensure unlinkability is sufficiently preserved (as seen by an attacker) between the partial identities of an individual person required by the applications	Employ privacy preserving identity management, e.g. proposed in [72], together with user-controlled identity management system [73] to ensure user-controlled linkability of personal data. System supports the user in making an informed choice of pseudonyms, representing his or her partial identities. Make the flow of this user's identity attributes explicit to the user and gives its user a large degree of control.
		Confidentiality of data flow in user-portal communication	Deploy anonymity system such as TOR [48].
7	Information disclosure at the social network data store	Release of the social network data store should be controlled according to user's privacy preference	Apply access control at the social network databases, e.g. privacy aware collaborative access control based on relationships [79]
8	Information disclosure of communication between the user and the social network	Confidentiality of communication between the user and the social network should be ensured	Employ a secure communication channel and deploy anonymity system such as TOR [48].
9	Content unawareness of user	Users need to be aware that they only need to provide minimal set of required personal data (the data minimization principle)	Use feedback tools to raise user's privacy awareness.
10	Policy and consent noncompliance of the whole social network system	Design system in compliance with legal guidelines for privacy and data protection	1) Hire employee who is responsible for making the policies compliant OR hire external company for compliancy auditing 2) Ensure training obligations for employees.
		Ensure user aware that in case of violation, user is legitimated to take legal actions	E.g., user can sue the social network provider whenever users personal data is not processed according to what is consented.
		Employee contracts clearly specify do's and don'ts according to legal guidance	1) Ensure training obligations for employees; 2) Employees who disclose users information will be penalized (get fired, pay fine, etc.).

problem to raise user's privacy awareness. The usability of such feedback tools is also an issue, such as how to design a user friendly interface and encourage users to use feedback remains a research challenge.

9 Related work

9.1 Privacy requirements analysis

Mylopoulos et al. [87] are the first to point out that complexity of an information system is determined partly by its functionality (i.e. what the system does) and partly by its non-functional requirements (also referred as constrains, goals, and quality attributes), and the non-functional requirements "play a crucial role during system development, serving as selection criteria for choosing among myriads of decisions". They proposed a comprehensive framework for representing and using non-functional requirements during the development process in a process-oriented approach. The framework consists of five basic components – goals, link types, methods (i.e., goal decomposition methods, goal satisficing methods, and argumentation methods), correlation rules, and the labeling procedure – as the representation of non-functional requirements in terms of interrelated goals. As suggested by Mylopoulos et al., "such goals can be refined through refinement methods and can be evaluated in order to determine the degree to which a set of non-functional requirements is supported by a particular design" [87]. This framework can serve as the foundation for the aforementioned privacy requirement analysis methodology.

The privacy guidelines provided by Microsoft describes some basic privacy concepts [88], such as different types of consents or data minimization concepts. Besides, a number of guidelines are presented for selected scenarios concerning the following principles: notice, choice, onward transfer, access, security, and data integrity. However, it only contains a flat list of the required and recommended guidelines and does not intend to describe a more structured approach. These guidelines can still be used as inspiration to determine possible threats, e.g. to extend our catalogue of threat trees.

In the documentation of SDL version 3.2 [89], privacy is also partially considered, but only at a generic level. For example, the threat modeling process described in the forth stage of SDL only mentions that a design review with the privacy expert is necessary. SDL also presents ten general privacy guidelines. An example guideline indicates that it is important to collect the least sensitive form of data. At this stage, privacy is not yet well integrated in SDL.

Yu and Cysneiros [90] presented a framework using $i*$, an agent-oriented requirements modeling language, to deal with privacy requirements. This framework however focuses on reasoning about privacy and does not provide a structured methodology to examine the different privacy objectives. Liu et al. [91] proposed a framework also using $i*$ to deal with security and privacy requirements, which was inspired by the work of Yu and Cysneiros. They use four different analysis techniques to create a complete model: attacker analysis, dependency vulnerability analysis, countermeasure analysis and access control analysis. Although this framework contains the attacker analysis technique which is similar to our idea of examining the possible privacy threats, the framework is again mainly meant to reason about privacy but lacks the necessary knowledge to empower the (non-expert) privacy analyst.

Miyazaki et al. [92] defined a computer-aided privacy requirements elicitation technique. This technique returns the appropriate requirements for the system to be compliant with the law, based on a questionnaire that the system engineer fills out. This is in contrast to our methodology, which helps the analyst eliciting the privacy requirements in accordance to the stakeholders' wishes.

9.2 From privacy requirements to privacy solutions

Several taxonomies have been proposed to create a link between privacy requirements and privacy solutions. This section gives an overview of some existing taxonomies which can be integrated in our threat modeling process to link the privacy requirements obtained by our methodology to the optimal privacy enhancing solution(s).

The taxonomy of privacy goals, described by Antón et al. [93], is to analyze website privacy requirements. Privacy goals are divided into protection goals and (anti) vulnerability goals. The Code for Fair Information Practices is used to categorize the protection goals, namely notice and awareness, choice and consent, access and participation, integrity and security, and enforcement and redress. The (anti) vulnerability goals are classified according to the manner in which they violate the users privacy. The corresponding goals are monitoring, aggregation, storage, and information transfer. These goals are used to analyze and compare privacy policies and do not intend to link privacy requirements to general privacy solutions, instead, limited to website privacy requirements.

The Pris method [35] presents a structured way to create systems which adhere to the specified privacy requirements. It consists of four different phases: 1) Elicit privacy-related goals, 2) Analyze the impact of privacy goals on organizational processes, 3) Model affected processes using privacy-process patterns, and 4) Identify the technique(s) that best support or implement the above processes. The last step uses a table that classifies privacy implementation techniques in six categories, a) Administrative tools, b) Information tools, c) Anonymizer products, services and architectures, d) Pseudonymizer tools, e) Track and evidence erasers, and f) En-

cryption tools, and it can be an interesting source of inspiration to determine appropriate privacy solutions.

Wuyts et al. [34] created a hierarchical taxonomy which categorizes privacy into objectives. The objectives are divided in two branches. The proactive branch focuses on concealing the association between the data and the identity of the user before it is shared with the system, while the reactive branch focuses on guarding the relationship between the data and the identity when the data is already shared. Each objective corresponds to a number of strategies, which are a sub-classification of the objectives. These strategies can then be linked to their corresponding solutions. This paper only provides a sample solution for each category and does not provide a full overview of all existing solutions.

10 Discussion

Despite the fact that the dualism between hard and soft privacy has already been generically introduced in a few talks [94, 12], to our best knowledge, this paper is the first effort that concretely distinguishes these two concepts and categorizes privacy properties (and threats) accordingly. In short, hard privacy properties include unlinkability, anonymity and pseudonymity, plausible deniability, undetectability and unobservability, and confidentiality. Soft privacy properties include content unawareness and policy and consent compliance. Similarly, leveraging the link between privacy enhancing technologies and privacy objectives, it is possible to make a distinction between hard and soft solutions. Hard privacy technologies are active in research but poor in deployment, due to cost and technical evolvement restrictions (such as cryptography). Soft privacy technologies are state-of-art and have fewer research activities. With legal compliance as a strong driver, soft privacy solutions reply on stakeholder's liability and the tradeoffs between cost of deploying privacy solutions and potential costs in case of massive data breach. After all, building in privacy in the system might not be cheap, but just cheaper than building in no privacy.

There are a number of insights concerning the proposed methodology that are worthy to be emphasized:

1. Some privacy threats, in contrast to security, affect DFD elements pair-wise (or sometimes group-wise). For instance, unlinkability implies the relation of two or more items of interest. As an example, a relation may refer to a subject (an entity in the DFD) and its attributes (in the data stores). Consequently, it is straightforward to see that linkability threats always affect a pair or a group of DFD elements. Similarly, plausible deniability refers to the pair-wise relation between a subject and the attribute that the subject wants to deny. We can draw a conclusion that privacy emphasizes relationships between instances of DFD elements (e.g. two communication instances of the same data flow cannot be linked) or relationships between a DFD element and an entity, while security focuses on each individual DFD component in a more local way.
2. The *process* element in the DFD is less important for privacy because privacy cares more about the relationships between entities and data. It is quite the opposite in the case of security. For instance, all STRIDE threat categories apply to the process element, while, in LINDDUN, all privacy attack paths involving the process element are related to a security threat (namely, information disclosure of process) and not to privacy threats per se. This observation leads to our next finding.
3. The authors have chosen the DFD notation to represent a software-based system in order to keep compliance with the STRIDE approach. As STRIDE and LINDDUN are expected to be applied in a synergic way, this choice fosters the reuse of the DFD models (and the modeling knowledge) across the security and the privacy threat analysis.
   However, it should be noticed that DFD elements (entities, processes, data flows, and data stores) represent the technical assets that require protection from privacy-specific harm. That is, the DFDs expose the privacy-relevant information concerning the technical assets from the system perspective. Therefore, the privacy analyst should pay attention to the fact that the important privacy assets are properly modeled by the DFD. For instance, data flows should be specific to IOIs in the privacy case.
4. For some privacy properties, an extension of the DFD semantics might be useful. For instance, for the case of privacy-sensitive information that is inferred over time, the notion of knowledge is necessary. This could be annotated in the DFD processes via epistemic constructs and then leveraged during the analysis.
5. Concerning the privacy threat trees, one can see that many paths lead to security threats (e.g. information disclosure and tampering). Consequently, privacy objectives heavily depend on security objectives (e.g. confidentiality and integrity of data flow, data store or process). We draw the general conclusion that it is interesting to analyze privacy threats together with security threats and the necessary process (and tool) support should be built to facilitate such activities. This synergy would bring an advantage in terms of time and cost for system designers to work with. Further, privacy and security objectives might conflict (e.g. non-repudiation and plausible deniability, as explained in the previous sections). It is thus useful to perform the threat analysis for privacy and security altogether and consider both types of requirements at the same time.

Finally, it is necessary to clarify a few definitions to avoid confusion. Confidentiality refers to hiding the data content; anonymity and pseudonymity refer to hiding the subject's identity or hiding the link between identity and action or a piece of information; unlinkability refers to hiding links between two or more objectives (actions, identities, and pieces of information); and undetectability and unobservability refers to hiding a subject's activity. In particular, unlinkability of entities means that it is impossible to relate different entities based on some common personal identifiable information (PII), while anonymity of entities means that the subject cannot be identified within a anonymity set. Anonymity at data flow typically means that, in an anonymous communication setting, the subject cannot be identified from the data flow's content or side channel information; while anonymity at data store refers to data anonymization, meaning that the subject cannot be identified from the content of the data store.

11 Conclusion

In this paper, we have presented a comprehensive framework to model privacy threats in software-based systems, elicit privacy requirements, and instantiate privacy enhancing countermeasures. The primary contribution is the systematic methodology to model privacy specific threats. This is achieved by defining a list of privacy threat types and providing the necessary mappings to the elements in the system model. The second contribution is represented by the supporting body of knowledge, namely, an extensive catalogue of privacy specific threat tree patterns. In addition, this work provides the means to map the most commonly known privacy enhancing technologies (PETs) to the identified privacy threats and the elicited privacy requirements. The privacy threat tree patterns and categorization of suggested PETs are expected to be continuously updated and improved upon, since new threats keep emerging, just as new privacy technologies keep evolving.

As future work, we plan to apply the proposed framework to larger case studies, for instance, validation in the context of a national e-health system is being performed.

References

1. A. van Lamsweerde, S. Brohez, R. D. Landtsheer, and D. Janssens, "From system goals to intruder anti-goals: Attack generation and resolution for security requirements engineering," in *Proceedings of the RE'03 Workshop on Requirements for High Assurance Systems (RHAS'03)*, pp. 49–56, 2003.
2. A. van Lamsweerde, *Requirements Engineering: From System Goals to UML Models to Software Specifications*. Wiley, 2009.
3. M. Howard and S. Lipner, *The Security Development Lifecycle*. Microsoft Press, 2006.
4. G. McGraw, *Software Security: Building Security In*. Addison-Wesley Professional, 2006.
5. B. Schneier, *Secrets & Lies: Digital Security in a Networked World*. Wiley, 2000.
6. G. S. Andreas and A. L. Opdahl, "Templates for misuse case description," in *Proceedings of the 7 th International Workshop on Requirements Engineering, Foundation for Software Quality*, pp. 4–5, 2001.
7. "Experimental comparison of attack trees and misuse cases for security threat identification," *Information and Software Technology*, vol. 51, no. 5, pp. 916–932, 2009. SPECIAL ISSUE: Model-Driven Development for Secure Information Systems.
8. D. J. Solove, "A taxonomy of privacy," *University of Pennsylvania Law Review*, vol. 154, January 2006. http://ssrn.com/abstract=667622 .
9. D. J. Solove, *Understanding Privacy*. Harvard University Press, May 2008.
10. A. Pfitzmann and M. Hansen, "A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management (Version 0.33 April 2010)," tech. rep., TU Dresden and ULD Kiel, April 2010. http://dud.inf.tu-dresden.de/Anon_Terminology.shtml .
11. M. Hansen, "Linkage control - integrating the essence of privacy protection into identity management systems," in *Collaboration and the Knowledge Economy: Issues, Applications, Case Studies* (P. Cunningham and M. Cunningham, eds.), Proceedings of eChallenges, pp. 1585–1592, IOS Press, Amsterdam, 2008.
12. G. Danezis, "Talk: an introduction to u-prove privacy protection technology, and its role in the identity metasystem – what future for privacy technology." http://www.petsfinebalance.com/agenda/index.php , 2008.
13. "ISO 17799: Information technology - code of practice for information security management," tech. rep., British Standards Institute, 2000.
14. M. Roe, *Cryptography and Evidence*. PhD thesis, University of Cambridge, Clare College, 1997.
15. E. McCallister, T. Grance, and K. Kent, "Guide to protecting the confidentiality of personally identifiable information (PII) (draft)," tech. rep., National Institute of Standards and Technology (U.S.), 2009.
16. S. Lederer, J. I. Hong, A. K. Dey, and J. A. Landay, "Personal privacy through understanding and action: Five pitfalls for designers," *Personal and Ubiquitous Computing*, vol. 8, pp. 440–454, 2004.
17. S. Patil and A. Kobsa, *Privacy Considerations in Awareness Systems: Designing with Privacy in Mind*, ch. 8, pp. 187–206. Human-Computer Interaction Series, Springer London, June 2009.
18. P3P, "Platform for privacy preferences project: W3C P3P specifications." http://www.w3.org/TR/P3P/ .
19. EU, "Directive 95/46/EC of the European parliament and of the council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data.," *Official Journal of the European Communities*, vol. 281, pp. 31–50, 1995. http://ec.europa.eu/justice_home/fsj/privacy/index_en.htm .
20. "HIPAA administrative simplification: enforcement; final rule. United States Department of Health & Human service.," *Federal Register / Rules and Regulations*, vol. 71, no. 32, 2006.
21. "PIPEDA – Personal information protection and electronic documents act (2000, c. 5)." http://laws.justice.gc.ca/en/showtdm/cs/P-8.6 , October 2009.
22. A. national privacy regulator, "Australia's national privacy regulator: Privacy act." http://www.privacy.gov.au/law/act .

23. OECD, "Guidelines on the protection of privacy and transborder flows of personal data, organization for economic cooperation and development." http://www.oecd.org/document/18/0,2340,en_2649_34255_1815186_1_1_1_1,00.html , 1980.
24. T. D. Breaux, A. I. Anton, K. Boucher, and M. Dorfman, "Legal requirements, compliance and practice: An industry case study in accessibility," in *RE'08: Proceedings of the 16th IEEE International Requirements Engineering Conference (RE'08)*, pp. 43–52, IEEE Society Press, September 2008.
25. U. D. of Justice, "Workforce Investment Act of 1998, SEC. 508. electronic and information technology." http://www.justice.gov/crt/508/508law.php .
26. T. Breaux and A. Antón, "Analyzing regulatory rules for privacy and security requirements," *IEEE Trans. Softw. Eng.*, vol. 34, no. 1, pp. 5–20, 2008.
27. G. Danezis, C. Diaz, and P. Syverson, *Systems for Anonymous Communication, in CRC Handbook of Financial Cryptography and Security*, p. 61. Chapman & Hall, 2009.
28. L. Sweeney, "K-anonymity: a model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, 2002.
29. I. Alexander, "Misuse cases: Use cases with hostile intent," *IEEE Software*, vol. 20, no. 1, pp. 58–66, 2003.
30. OWASP, "Risk rating methodology." http://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology .
31. M. Library, "Improving web application security: Threats and countermeasures."
32. NIST, "Risk management guide for information technology systems, special publication 800-30." http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf .
33. C. S. E. Institute, "OCTAVE – (Operationally Critical Threat, Asset, and Vulnerability Evaluation)." http://www.cert.org/octave/ .
34. K. Wuyts, R. Scandariato, B. D. Decker, and W. Joosen, "Linking privacy solutions to developer goals," in *Proceeding of the International Conference on Availability, Reliability and Security*, pp. 847–852, IEEE Computer Society, 2009.
35. C. Kalloniatis, E. Kavakli, and S. Gritzalis, "Addressing privacy requirements in system design: the pris method," *Requirements Engineering*, vol. 13, pp. 241–255, September 2008. http://dx.doi.org/10.1007/s00766-008-0067-3 .
36. "PETs – annual symposium on privacy enhancing technologies homepage." http://petsymposium.org/ .
37. D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–88, 1981.
38. D. Chaum, "Security without identification: Transaction systems to make big brother obsolete," *Communications of the ACM*, vol. 28, no. 10, pp. 1030–1044, 1985.
39. D. Chaum, "The dining cryptographers problem: Unconditional sender and recipient untraceability," *Journal of Cryptology*, vol. 1, no. 1, pp. 65–75, 1988.
40. A. Pfitzmann, B. Pfitzmann, and M. Waidner, "ISDN-mixes: Untraceable communication with very small bandwidth overhead," in *Proceedings of the GI/ITG Conference on Communication in Distributed Systems*, pp. 451–463, February 1991.
41. D. M. Goldschlag, M. G. Reed, and P. F. Syverson, "Hiding Routing Information," in *Proceedings of Information Hiding: First International Workshop* (R. Anderson, ed.), pp. 137–150, Springer-Verlag, LNCS 1174, May 1996.
42. M. Reiter and A. Rubin, "Crowds: Anonymity for web transactions," *ACM Transactions on Information and System Security*, vol. 1, pp. 66–92, June 1998.
43. A. Bacard, "Anonymous.to: Cypherpunk tutorial." http://www.andrebacard.com/remail.html .
44. "Mixmaster." http://mixmaster.sourceforge.net/ .
45. "Mixminion." http://mixminion.net/ .
46. A. Back, I. Goldberg, and A. Shostack, "Freedom systems 2.1 security issues and analysis," white paper, Zero Knowledge Systems, Inc., May 2001. http://www.cypherspace.org/adam/pubs/freedom-21.pdf .
47. O. Berthold, H. Federrath, and S. Köpsell, "Web MIXes: A system for anonymous and unobservable Internet access," in *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability* (H. Federrath, ed.), LNCS 2009, pp. 115–129, Springer-Verlag, July 2000.
48. T. project, "Tor: anonymity online." http://www.torproject.org/ .
49. A. Pfitzmann and M. Waidner, "Networks without user observability – design options," in *Advances in Cryptology - EUROCRYPT'85*, LNCS 219, pp. 245–253, Springer-Verlag, 1985.
50. M. Waidner and B. Pfitzmann, "The dining cryptographers in the disco: Unconditional sender and recipient untraceability," in *Advances in Cryptology - EUROCRYPT'91*, LNCS 434, p. 23, Springer-Verlag, 1990.
51. M. Abadia and C. Fournet, "Private authentication," *Theoretical Computer Science*, vol. 322, pp. 427–476, September 2004.
52. W. Aiello, S. M. Bellovin, M. Blaze, R. Canetti, J. Ioannidis, A. D. Keromytis, and O. Reingold, "Just fast keying: Key agreement in a hostile internet," *ACM Trans. Inf. Syst. Secur*, vol. 7, p. 2004, 2004.
53. S. Brands and D. Chaum, "Distance-bounding protocols (extended abstract)," in *Advances in Cryptology - EUROCRYPT'93*, LNCS 765, pp. 344–359, Springer-Verlag, 1993.
54. J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Advances in Cryptology - CRYPTO'04*, LNCS 3152, pp. 56–72, Springer-Verlag, 2004.
55. M. Naor, "Deniable ring authentication," in *Advances in Cryptology - CRYPTO 2002*, LNCS 2442, pp. 481–498, Springer-Verlag, 2002.
56. N. Borisov, I. Goldberg, and E. Brewer, "Off-the-record communication, or, why not to use pgp," in *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, pp. 77–84, ACM, 2004.
57. A. C.-C. Yao, "Protocols for secure computations," in *Proceedings of Twenty-third IEEE Symposium on Foundations of Computer Science*, pp. 160–164, November 1982.
58. M. Naor and K. Nissim, "Communication complexity and secure function evaluation," *CoRR*, vol. cs.CR/0109011, 2001.
59. M. Deng, T. Bianchi, A. Piva, and B. Preneel, "An efficient buyer-seller watermarking protocol based on composite signal representation," in *Proceedings of the 11th ACM workshop on Multimedia and security*, (Princeton, New Jersey, USA), pp. 9–18, ACM New York, NY, USA, 2009.
60. B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Journal of the ACM*, pp. 41–50, 1998.
61. M. O. Rabin, "How to exchange secrets by oblivious transfer," technical report TR-81, Aiken Computation Laboratory, Harvard University, 1981.
62. C. Cachin, "On the foundations of oblivious transfer," in *Advances in Cryptology – Eurocrypt '98*, LNCS 1403, pp. 361–374, Springer-Verlag, 1998.
63. V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-art in privacy preserving data mining," *ACM SIGMOD Record*, vol. 3, pp. 50–57, Mar. 2004.
64. B. Pinkas, "Cryptographic techniques for privacy preserving data mining," *SIGKDD Explorations*, vol. 4, no. 2, pp. 12–19, 2002.
65. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-lee, G. Neven, Pascal, P. Paillier, and H. Shi, "Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions," in *Advances in Cryptology - CRYPTO'05*, LNCS, pp. 205–222, Springer-Verlag, 2005.

66. R. Ostrovsky and W. E. S. III, "Private searching on streaming data," in *Advances in Cryptology - CRYPTO'05*, LNCS, pp. 223–240, Springer-Verlag, 2005.
67. L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 571–588, 2002.
68. A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "*l*-diversity: Privacy beyond *k*-anonymity," in *International Conference on Data Engineering (ICDE'06)*, p. 24, IEEE Computer Society, 2006.
69. R. Anderson and F. Petitcolas, "On the limits of steganography," *IEEE Journal of Selected Areas in Communications*, vol. 16, pp. 474–481, 1998.
70. I. Moskowitz, R. E. Newman, D. P. Crepeau, and A. R. Miller, "Covert channels and anonymizing networks," in *In Workshop on Privacy in the Electronic Society*, pp. 79–88, ACM, 2003.
71. D. Kirovski and H. S. Malvar, "Robust covert communication over a public audio channel using spread spectrum," in *Information Hiding*, LNCS, pp. 354–368, Springer-Verlag, 2001.
72. M. Hansen, P. Berlich, J. Camenisch, S. Clauß, A. Pfitzmann, and M. Waidner, "Privacy-enhancing identity management," *Information Security Technical Report (ISTR)*, vol. 9, no. 1, pp. 35–44, 2004. http://dx.doi.org/10.1016/S1363-4127(04)00014-7 .
73. S. Clauß, A. Pfitzmann, M. Hansen, and E. V. Herreweghen, "Privacy-enhancing identity management." The IPTS Report 67, pages 8-16, September 2002.
74. K. Simoens, P. Tuyls, and B. Preneel, "Privacy weaknesses in biometric sketches," in *Proceedings of the 2009 30th IEEE Symposium on Security and Privacy*, pp. 188–203, IEEE Computer Society, 2009.
75. A. J. Menezes, P. C. V. Oorschot, S. A. Vanstone, and R. L. Rivest, *Handbook of Applied Cryptography*. CRC Press, December 1996. http://www.cacr.math.uwaterloo.ca/hac/ .
76. C. Fontaine and F. Galand, "A survey of homomorphic encryption for non-specialists," *EURASIP Journal on Information Security special issue Signal Processing in the Encrypted Domain*, p. Article ID 13801, 2007.
77. J. Camenisch and I. Damgård, "Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes," in *Advances in Cryptology - ASIACRYPT 2000*, LNCS, pp. 331–345, Springer-Verlag, 2000.
78. C. K. Georgiadis, I. Mavridis, G. Pangalos, and R. K. Thomas, "Flexible team-based access control using contexts," in *ACM symposium on Access control models and technologies*, pp. 21–27, ACM, 2001.
79. B. Carminati and E. Ferrari, "Privacy-aware collaborative access control in web-based social networks," in *Proc. of the 22nd IFIP WG 11.3 Working Conference on Data and Applications Security (DBSEC2008)*, pp. 81–96, Springer Berlin / Heidelberg, July 2008.
80. C. A. Ardagna, J. Camenisch, M. Kohlweiss, R. Leenes, G. Neven, B. Priem, P. Samarati, D. Sommer, and M. Verdicchio, "Exploiting cryptography for privacy-enhanced access control: A result of the PRIME project," *Journal of Computer Security*, vol. 18, pp. 123–160, January 2010.
81. OASIS, "eXtensible access control markup language: XACML 3.0." http://xml.coverpages.org/xacml.html .
82. IBM, "Enterprise privacy authorization language: EPAL 1.2." http://www.w3.org/Submission/2003/SUBM-EPAL-20031110/ .
83. H. R. Lipford, A. Besmer, and J. Watson, "Understanding privacy settings in facebook with an audience view," in *Proceedings of the 1st Conference on Usability, Psychology, and Security* (E. F. Churchill and R. Dhamija, eds.), USENIX Association Berkeley, CA, USA, 2008. http://www.usenix.org/events/upsec08/tech/full_papers/lipford/lipford.pdf .
84. J. Anderson, C. Diaz, J. Bonneau, and F. Stajano, "Privacy-enabling social networking over untrusted networks," in *Proceedings of the 2nd ACM workshop on Online social networks*, pp. 1–6, ACM, 2009.
85. F. Beato, M. Kohlweiss, and K. Wouters, "Enforcing access control in social networks." HotPets, 2009. http://www.cosic.esat.kuleuven.be/publications/article-1240.pdf .
86. "PrimeLife – the European PrimeLife research project – privacy and identity management in Europe for life." http://www.primelife.eu/ .
87. J. Mylopoulos, L. Chung, and B. Nixon, "Representing and using non-functional requirements: A process-oriented approach," *IEEE Transactions on Software Engineering*, vol. 18, pp. 483–497, 1992.
88. "Privacy guidelines for developing software products and services (version 3.1)," tech. rep., Microsoft Coorporation, September 2008.
89. "The security development lifecycle (SDL) version 3.2," tech. rep., Microsoft Coorporation, April 2008. http://www.microsoft.com/Downloads/details.aspx?familyid=2412C443-27F6-4AAC-9883-F55BA5B01814&displaylang=en.
90. E. Yu and L. M. Cysneiros, "Designing for privacy and other competing requirements," in *Proceedings of the 2nd Symposium on Requirements Engineering for Information Security*, pp. 15–16, 2002.
91. L. Liu, E. Yu, and J. Mylopoulos, "Security and privacy requirements analysis within a social setting," in *Proceeding of the IEEE International Conference on Requirements Engineering*, pp. 151–161, IEEE Computer Society, 2003.
92. S. Miyazaki, N. Mead, and J. Zhan, "Computer-aided privacy requirements elicitation technique," *Asia-Pacific Conference on Services Computing. 2006 IEEE*, pp. 367–372, 2008.
93. A. I. Antón, J. B. Earp, and A. Reese, "Analyzing website privacy requirements using a privacy goal taxonomy," in *RE'02: Proceedings of the 10th Anniversary IEEE Joint International Conference on Requirements Engineering*, pp. 23–31, IEEE Computer Society, 2002.
94. G. Danezis, "Talk: Introduction to privacy technology." http://research.microsoft.com/en-us/um/people/gdane/talks/Privacy_Technology_cosic.pdf , 2007.

A Misuse case examples

MUC 2: Linkability of of the user-portal data stream (data flow)

Summary: Data flows can be linked to the same person (without necessarily revealing the persons identity)
Asset: PII of the user

- The user:
  - data flow can be linked to each other which might reveal the persons identity
  - the attacker can build a profile of a user's online activities (interests, active time, comments, updates, etc.)

Primary misactor: skilled insider / skilled outsider
Basic Flow:

1. The misactor intercepts / eavesdrops two or more data flows
2. The misactor can link the data flows to each other and possibly link them (by combining this information) to the user / data subject

Trigger: by misactor, can happen whenever data is communicated
Preconditions:

- No anonymous communication system used
- Information disclosure of data flow possible

Prevention capture points:

- Use strong anonymous communication techniques
- Provide confidential channel

Prevention guarantee: Impossible to link data to each other

MUC 3: Linkability of the social network users (entity)

Summary: Entities (with different pseudonyms) can be linked to the same person (without necessarily revealing the persons identity)
Asset: PII of the user

- The user:
  - data can be linked to each other which might reveal the persons identity
  - attacker can build a profile of a user's online activities (interests, actives time, comments, updates, etc.)

Primary misactor: skilled insider / skilled outsider
Basic Flow:

1. The misactor intercepts or eavesdrops two or more pseudonyms
2. The misactor can link the pseudonyms to each other and possibly link (by combining this information) to the user / data subject

Trigger: by misactor, can happen whenever data is communicated
Preconditions:

- Information Disclosure of the data flow possible
- Different "pseudonyms" are linked to each other based on content of the data flow

Prevention capture points:

- protection of information such as user temporary ID, IP address, time and location, session ID, identifier and biometrics, computer ID, communication content, e.g. apply data obfuscation to protection this information (security)
- message and channel confidentiality provided

Prevention guarantee: Impossible to link data to each other

MUC 4: Identifiability at the social network database (data store)

Summary: The users identity is revealed
Asset: PII of the user

- The user: revealed identity

Primary misactor: skilled insider / skilled outsider
Basic Flow:

1. The misactor gains access to the database
2. The data is linked to a pseudonym
3. The misactor can link the pseudonym to the actual identity (identifiability of entity)
4. The misactor can link the data to the actual user's identity

Alternative Flow:

1. The misactor gains access to the database
2. The can link information from the database to other information (from another database or information which might be publicly accessible)
3. The misactor can re-identify the user based on the combined information

Trigger: by misactor, can always happen
Preconditions:

- no or insufficient protection of the data store
- no data anonymization techniques used

Prevention capture points:

- protection of the data store (security)
- apply data anonymization techniques

Prevention guarantee: hard-impossible to link data to identity (depending on applied technique)

MUC 5: Identifiability of user-portal data stream (data flow)

Summary: The users identity is revealed
Asset: PII of the user

- The user: revealed identity

Primary misactor: insider / outsider
Basic Flow:

1. The misactor gains access to the data flow
2. The data contains personal identifiable information about the user (user relationships, address, etc.)
3. The misactor is able to extract personal identifiable information from the user / data subject

Trigger: by misactor, can happen whenever data is communicated
Preconditions:

- no or weak anonymous communication system used
- Information disclosure of data flow possible

Prevention capture points:

- apply anonymous communication techniques
- Use confidential channel

Prevention guarantee: hard-impossible to link data to identity (depending on applied technique)

MUC 6: Identifiability of users of the social network system (entity)

Summary: The users identity is revealed
Asset: PII of the user

- The user: revealed identity

Primary misactor: skilled insider / skilled outsider
Basic Flow:

1. The misactor gains access to the data flow
2. The data contains the user's password
3. The misactor has access to the identity management database
4. The misactor can link the password to the user

Alternative Flow:

1. The misactor gains access to the data flow
2. The data contains the user's password
3. The misactor can link the user's password to the user's identity (password is initials followed by birthdate)

Trigger: by misactor, can happen whenever data is communicated and the user logs in using his "secret"
Preconditions:

- Insecure IDM system OR
- weak passwords used and information disclosure of data flow possible

Prevention capture points:

- Strong pseudonymity technique used (e.g. strong passwords)
- privacy-enhancing IDM system
- Data flow confidentiality

Prevention guarantee: hard(er) to link log-in to identity.

MUC 7: Information Disclosure at the social network database (data store)

Summary: Data is exposed to unauthorized users
Asset: PII of the user

- The user: revealed sensitive data

Primary misactor: skilled insider / skilled outsider
Basic Flow:

1. The misactor gains access to the database
2. The misactor retrieves data to which he should not have access

Trigger: by misactor, can always happen
Preconditions:

- no or insufficient internal access policies

Prevention capture points:

- strong access control policies (security). For example, rule-based access control based on friendships in the social network

Prevention guarantee: hard-impossible to obtain data without having the necessary permissions

MUC 8: Information Disclosure of communication between the user and the social network (data flow)

Summary: The communication is exposed to unauthorized users
Asset: PII of the user

- The user: revealed sensitive data

Primary misactor: skilled insider / skilled outsider
Basic Flow:

1. The misactor gains access to the data flow
2. The misactor retrieves data to which he should not have access

Trigger: by misactor, can happen whenever messages are being sent
Preconditions:

- communication goes through insecure public network

Prevention capture points:

- messages sent between user and social network web client is encrypted and secure communication channel is ensured

Prevention guarantee: hard-impossible to gain access to the data flow without having the right permissions

MUC 9: Content unawareness

Summary: User is unaware that his or her anonymity is at risk due to the fact that too much personal identifiable information is released
Asset: PII of the user

- The user: revealed identity

Primary misactor: skilled insider / skilled outsider
Basic Flow:

1. The misactor gain access to user's online comments
2. The misactor profiles the user's data and can identify the user

Trigger: by misactor, can always happen
Preconditions:

- User provides too much personal data

Prevention capture points:

- User provides only minimal set of required information

Prevention guarantee: user will be informed about potential privacy risks

MUC 10: Policy and consent noncompliance

Summary: The social network provider doesn't process user's personal data in compliance with user consent, e.g., disclose the database to third parties for secondary use
Asset: PII of the user

- The user: revealed identity and personal information
- The system / company: negative impact on reputation

Primary misactor: Insider
Basic Flow:

1. The misactor gains access to social network database
2. The misactor discloses the data to a third party

Trigger: by misactor, can always happen
Preconditions:

- misactor can tamper with privacy policies and makes consents inconsistent OR
- policies not managed correctly (not updated according to user's requests)

Prevention capture points:

- Design system in compliance with legal guidelines for privacy and data protection and keep internal policies consistent with policies communicated to user
- Legal enforcement: user can sue the social network provider whenever his or her personal data is processed without consents
- Employee contracts: employees who share information with 3th parties will be penalized (fired, pay fine, etc.)

Prevention guarantee: Legal enforcement will lower the threat of an insider leaking information but it will still be possible to breach user's privacy