Towards a Model of User-centered Privacy Preservation

Paul Grace
IT Innovation, University of Southampton
University of Southampton
Southampton, UK
pjg@it-innovation.soton.ac.uk

Mike Surridge
IT Innovation
University of Southampton
Southampton, UK
ms@it-innovation.soton.ac.uk

ABSTRACT

The growth in cloud-based services tailored for users means more and more personal data is being exploited, and with this comes the need to better handle user privacy. Software technologies concentrating on privacy preservation typically present a one-size fits all solution. However, users have different viewpoints of what privacy means to them and therefore, configurable and dynamic privacy preserving solutions have the potential to create useful and tailored services without breaching any user's privacy. In this paper, we present a model of user-centered privacy that can be used to analyse a service's behaviour against user preferences, such that a user can be informed of the privacy implications of that service and what fine-grained actions they can take to maintain their privacy. We show through a case-study that the user-based privacy model can: i) provide customizable privacy aligned with user needs; and ii) identify potential privacy breaches.

CCS CONCEPTS

•Security and privacy → Domain-specific security and privacy architectures; •Social and professional topics → Privacy policies; •Computer systems organization → *Cloud computing;*

KEYWORDS

Privacy, Cloud Computing, Model-driven development

ACM Reference format:
Paul Grace and Mike Surridge. 2017. Towards a Model of User-centered Privacy Preservation. In *Proceedings of ARES '17, Reggio Calabria, Italy, August 29-September 01, 2017,* 8 pages.
DOI: 10.1145/3098954.3104054

1 INTRODUCTION

The increasing growth in cloud-based services driven by the *-as-a-service model, means that an increasing amount of data (about the user) is being collected, stored and utilized. Already, users have a limited understanding of what personal data is leveraged by these online services, and for what purpose. Studies have shown that where services provide a privacy policy statement, users do not have the time to fully read and understand such policies, nor can they answer simple questions about what these policies mean [21]. Such privacy concerns will be further amplified by the growth of highly-complex Cloud, Big Data, Social Networks and IoT technologies centered upon personal data.

Furthermore, users have different viewpoints of what privacy means to them. Westin's privacy indexes provide evidence of this [15], i.e. one user may care about keeping a piece of data private, whereas another user may not care if the same data is made public. Many privacy-preservation solutions focus on a one-size-fits-all approach [9], [27] (e.g. to hide everything), but this approach to privacy fails to meet user requirements which also vary according to the data and purpose. For example, users may wish to disclose sensitive medical information to support medical research studies, but not allow an environmental monitoring service to track their movements. Current approaches to preserve privacy using policy enforcement [3] or privacy matching [19] do not adequately address the requirements for users to understand the privacy implications of how their data is being used, nor do they provide the ability for the user to to determine for themselves what information is disclosed to others, and when and how such information is communicated.

In this paper, we present a general purpose **user-centric privacy model** specified as a Labeled Transition System (LTS). Here, actions on personal data (e.g., a third-party reads personal data) model the change of a user's state of privacy. Analysis of this model can then determine if actions carried out by an online service conflict with a user's privacy preferences. Therefore, such a model provides two main benefits: *abstraction*: the ability to abstract a complex system's behaviour with regards to personal data and then machine analyze the privacy implications, and *re-usability*: the ability to use the model to analyze a wide range of policies using different analysis algorithms.

The key contributions of this paper are the specification of this user-centric privacy model of dynamic cloud-based services that can be used to perform:

- *Privacy implication analysis*: the service's usage of private data can be analyzed against a user's privacy preferences to determine the risk of a privacy breach. From this, the implications can be presented to the user along with recommendations on how to manage the risk.
- *Fine-grained privacy aware access control*: a cloud-based service's access control policies can be adapted to match user privacy preferences, blocking some

ARES '17, Reggio Calabria, Italy
978-1-4503-5257-4/17/08...$15.00
DOI: 10.1145/3098954.3104054

service actions (where these are not critical to the service operation). We present an algorithm to generate the user privacy model (as an LTS) for a service, and identify transitions that should be blocked such that service actions that pose too great a risk to privacy become inaccessible.

Structure. The remainder of this paper is structured as follows. Section 2 presents an analysis of the related work, and identifies the requirements for user-centric privacy preservation. Section 3 introduces the user-centric privacy model. Section 4 documents how the policy computation algorithm works. In Section 5, we evaluate the contributions using a case study approach. Finally, in section 6 we conclude the paper and explore the future directions of this research.

2 RELATED WORK

Policy based authorization: is the means by which a trusted cloud provider or online service provider (OSP) protects the privacy of users' data by allowing the user to set their own privacy policies, and then enforce them so that no unauthorized access is allowed [3]. However, it is not always the case that services allow users to customize policies; rather they employ their own privacy policies that may or may not match with a users preferences, and hence there is a need to analyze the extent to which there is agreement.

Many languages are available to specify privacy policies. Ponder [8] and XACML[1] are general purpose access control languages, whereas UMA[2] is an OAuth-based web-based access management protocol to coordinate protection and sharing of web resources that are under that user's control. The Platform for Privacy Preferences (P3P)[3] is a W3C specification that enables Web Sites to express their privacy policy in a machine readable standard, which is interpreted by agents installed in the browser so that the user does not need to read the policy of each site they visit. P2U [10] is a privacy policy language inspired by P3P but adapted to user information sharing in a secondary context; applications can offer and negotiate user data sharing with other applications according to an explicit user-editable privacy policy.

Other languages allow users to declare their preferences for the collection, usage, and storage of their data. APPEL [5] is the language prescribed by the W3C for describing user preferences, as a complement to P3P for describing service requirements. Whenever a user connects to a web site, the user's privacy agent checks that the user's APPEL policy matches the web sites stated P3P-privacy policy. Rei [11] has been proposed as a more expressive language [13] that could replace APPEL; it expresses policies over domain-specific ontologies described in RDF and OWL.

Privacy Policy Evaluation; a system's behaviour should be matched against both it's own privacy policy, and the stated preference of the user. Chinosi et al. [4] model a system's behaviour in terms of a Business Processing Model Notation (BPMN) diagram and then the goal is to check whether this is compliant with the system's P3P privacy policy. Short et al. [25] integrate links to the privacy policy in the system's workflow (e.g. the BPEL specification), these are then checked by an analysis tool at design time to determine if the workflow agrees with the policy. Li et al. 2006 [17] provide a similar method; rather than having a designer merge the workflow and policy, the approach converts both models (a BPEL specification and P3P policy) into a graph representation before formally analyzing the correctness of the graph. There are further model-based checking approaches that match behaviour against policy [24]. However, all of these solutions only check if a system behaves according to its stated privacy policy; there has been limited research into the evaluation of whether a system complies with a user's stated preferences. One approach by Lu et al. [19] converts a BPEL specification of a system into an interface automata and maps the users privacy requirements into Linear Temporal Logic statements. Using the Spin model checker, the conflicts between the system and user preferences are identified. The semantic policy framework KAoS [1], based on the OWL semantic web language, allows general purpose reasoning to be used in policy decisions or to detect policy conflicts.

Analysis. UMA and XACML are the most flexible and dynamic solutions currently in practice; they have the capability to define flexible fine-grained access control over distributed systems. However, the complexity of the policies may mean it is difficult to determine implications of these against user preferences. Technologies based around P3P underpin automated technologies to analyze and match policies against preferences (removing the difficulties of understanding policies directly). However, P3P has also been criticized for being too complicated to accurately express policies [6]; further, P3P is not well-suited to match policies of dynamic cloud based services with multiple stakeholders who use and share personal data. These solutions generally seek to describe and analyze policies for a single system, or at best a static configuration of components. They cannot easily be used to determine policy implications in dynamic systems where the set of users and online services, their requirements and assurances are all in principle dynamic. Efforts to model the dynamic creation and propagation of access rights include the Take-Grant protection model [18], and capability-based approaches [22] that typically leverage CSP methods. Tools are also available such as Scollar [26] and SAM [16], which can check that certain goals are met (e.g. A never gets access to B, even after dynamic changes permitted by the policy). While formal approaches can assess the safety of a policy system; such methods are at odds with user-oriented, and comprehensible tools to understand privacy decisions and implications. Therefore, the user-centric privacy model in this paper models behavior centered upon the user, allowing analysis of complex policies against privacy preferences in order to determine where privacy breaches occur.

[1] http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html
[2] https://docs.kantarainitiative.org/uma/rec-uma-core.html
[3] https://www.w3.org/P3P/

Privacy Implications; in order for users to manage their usage of a system (whether that is in terms of changing the privacy settings, customizing the privacy policy, or altering their own interactions) they must be able to visualize and comprehend the implications of the system's privacy stance. Privacy Bird [7] was an Internet Explorer plug-in that was built around P3P policies and preferences. A bird was displayed in the corner of the browser and changed color when the user visited a site in conflict with the users preferences. Terms of Service; didnt read[4] provides crowd-sourced analysis of the privacy policies of well-known Internet Services e.g. Facebook and Google. The collective performs the analysis and gives the service a classification (e.g. A is good, E is poor) that is displayed when the user visits the site. Privacy Nutrition Labels [12] are inspired by food nutrition labels and capture the most important privacy elements from a systems policy; colour intensity is used to determine the level of privacy threats (i.e. red is bad, green is good). Human classification plays an important role in these systems; however, current research is also investigating how to leverage natural language processing to semi-automate the analysis of privacy policies to create the privacy nutrition label [23].

Analysis. Apart from Privacy Bird, these techniques dont consider the implications with regards to the individual user's preferences; and there is evidence suggesting users want finer-grained explanations about privacy implications in a medium easily understandable to them [14]. Research in the field of social networks highlights this requirement. PViz [20] is a tool that shows to users how their social groupings and chosen privacy policies affect their privacy, i.e. they can view who can see their profile. Privacy Nudges for Facebook [28] is an evaluation tool, which for each action the user could take (e.g. post a photo, or update status) displays the implication of that action to the user, e.g. *"this post will be seen by X and Y: do you want to do that?"*. The user-centric privacy model in this paper seeks to provide the output of privacy analysis to inform each user's fine-grained privacy implications.

Requirements. To go beyond the state of the art, a user-centric privacy preservation technology must provide: i) fine-grained control of an online service's usage of their data; ii) analysis of whether user preferences match the privacy stance of a service; and iii) understanding of the implications of usage of a service with respect to user privacy. Existing solutions do not attempt to resolve all of these requirements in an end-to-end manner, nor do they consider the dynamic nature of changing systems. In this paper, we present a formal model that will model a user's privacy in a dynamic system, closely aligning it with the user's privacy preferences. Analysis of the model can then be used to support the generation of privacy policies supporting fine-grained access control. The model can also underpin the interpretation of privacy implications, whereby the model is transformed into a simpler form and displayed to the user to help explain privacy decisions.

3 MODELING USER PRIVACY

3.1 Background

A user's privacy is changed by *actions* performed on *personal data* by both the user and other parties. Here, for the purposes of this research we recognize the other parties as: an online service provider, plus other service providers, and individuals who use personal data supplied to the online service with different roles and reasons. An *actor* within the service may read the personal data, may disclose it to another person, may process the data and update its value. Hence, each of these actions will change the user's privacy state in ways they may or may not find acceptable. Such behaviour can be formally modelled using a Finite State Machine (FSM); this is built upon the hypothesis of Kosa [14]: *We theorize that all data subjects have a current state of privacy at a point of time, and the sharing of personal information causes a change in that privacy state.*

Kosa's hypothesis was realized as an FSM model of user privacy; transitions represent events on personal data e.g. a user discloses personal information about themselves or their property. Hence, each event changes the state of a user's privacy. Kosa's model is hand-created by a privacy expert to model the system's behaviour, and can then be used to evaluate regulation compliance (again with human intervention). Such a model is a useful starting point, but it does not model the concrete actions of online services in detail (instead it focuses on the higher abstraction of disclosure events). In that regard, the model cannot analyze the impact of a fine grained action, e.g. Service user A with Role B using function E discloses Data Field D to service user C.

We therefore applied and extended this approach to online service usage of a user's data in order to capture privacy states (the user's state of privacy) based upon user-preferences. For example, when an event transitions to a general privacy state in Kosa's approach, our model transitions to a state that reflects the user's preference and risk for such disclosure. Our privacy model thereby adds two novel contributions:

- It can be analyzed for multiple purposes such as privacy policy computation, informing users of privacy implications by services, and regulation compliance, while Kosa [14] considers only compliance with legal regulations;
- It can be used to autonomously evaluate the models of service behaviour and access control and compute user-centric actions that control the privacy behaviour of the service in order to respect the user's wishes.

3.2 An LTS model of Privacy

A labeled transition system (LTS) is used to describe the potential behaviour of discrete systems. It is composed of the key elements of `states`, and `transitions` between states. Transitions are labeled with labels that correspond to discrete events that may occur during the operation of a system. We propose an LTS Model of an individual user's privacy

[4] http://tosdr.org

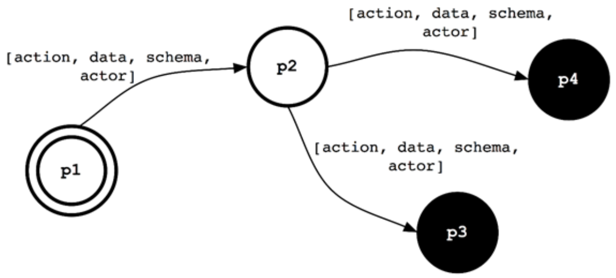

[action, data, schema, actor]
[action, data, schema, actor]
[action, data, schema, actor]
p1
p2
p3
p4

Figure 1: The general LTS model of a users privacy within a services behaviour

(in terms of their privacy preferences) with regards to the system behaviour of one or more interacting online services. A general illustration of this LTS model is shown in Figure 1.

The **User-Centric Privacy Model** is formed of the following the key elements:

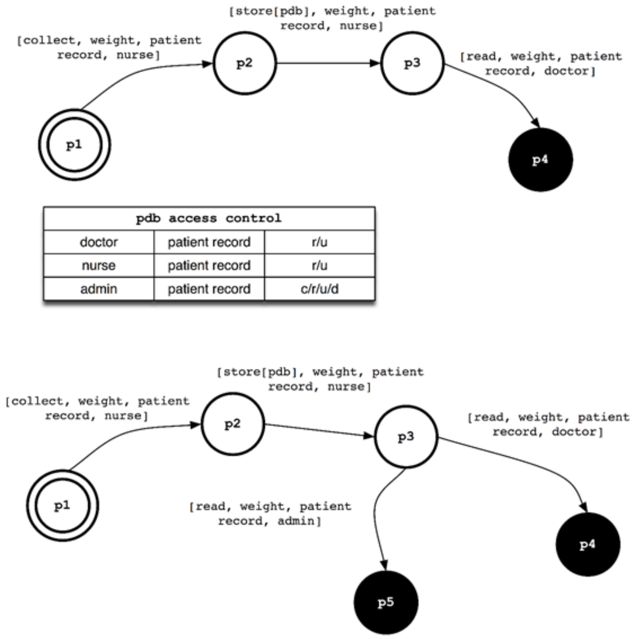
- `States`: a state is a representation of a user's privacy (it is not the state of a system or service). This evaluates to one of two values `agree, conflict`. Where conflict represents that the user's privacy is no longer in-line with their privacy preferences. There are three types of state in the model: i) a `start` state (`p1` in Figure 1) in which the user's data is always private; ii) `normal` states (`p2` in Figure 1) which have one and only one incoming transition; and iii) `end` states (`p3` and `p4` in Figure 1) which have no outgoing transitions. All states (except `end` states) have one or more outgoing transitions.
- `Transitions`: are each labeled as `action, data, data schema, actor`. These describe the actions that are carried out on personal data by a particular actor within the online service. An `actor` can be defined in terms of a role, i.e. doctor, nurse, admin, pharmacist in a medical service, or in terms of actor id e.g. Alice, Bob, or Dr Smith (n.b., such an individual actor may be a person, institute, company etc.). `Data` is a piece of data within a data element whose format and meta-data is provided by the `data schema`. Some example actions are listed in Table 1.

3.3 Creating the User Privacy Model

Figure 2 highlights the process required to create an LTS model of the user's privacy for a given online service. This is performed in two steps:

(1) *Create a model* of the online service's behaviour. Here, the service is broken down into the sequence of actions that are carried out by actors of the service. These correspond to the transitions in the LTS model. For example, in Figure 2 the workflow is a simple sequence of actions from state `p1` to `p4`: the user's weight is collected (measured) by a nurse, who stores this weight in the patient record in the patient database system (`pdb`), and a doctor reads the weight of the user from the patient record. Actions from the workflow lead to a set of **necessary** transitions in the corresponding LTS model of the service.

(2) *Compute potential actions* from access control. The access control list (e.g. for the `pdb` access control table in Figure 2) is also an input to the model which specifies actions that are allowed by the service. Access control policies generally restrict the possible actions of each type of user, but usually they do allow some actions that may not be present in the intended service workflow. For example, the admin actor in Figure 2 has full create, read, update and delete (CRUD) access to the patient record in the pdb database. This means an extra transition is needed in Figure 2 from `p3` to `p5` whereby admin reads the user's weight. This is not a mandatory transition, i.e. the service can function without it, but it is still possible and may or may not be compatible with the user's privacy preferences.


[collect, weight, patient record, nurse]
[store[pdb], weight, patient record, nurse]
[read, weight, patient record, doctor]
p1
p2
p3
p4
[collect, weight, patient record, nurse]
[store[pdb], weight, patient record, nurse]
[read, weight, patient record, doctor]
[read, weight, patient record, admin]
p1
p2
p3
p4
p5

pdb access control		
doctor	patient record	r/u
nurse	patient record	r/u
admin	patient record	c/r/u/d

Figure 2: The process to generate and LTS model from the workflow and access control

At this point, we have an LTS model of the service applicable to all users. That is, the state labels `p1` to `p5` are generic labels and we need to dynamically compute the user-specific privacy state values for each individual user based upon their preferences.

3.4 User Privacy Preferences

Users have different attitudes towards privacy. This has been established by a significant amount of research; for example, Westin carried out privacy surveys over a period of 30 years [15]. These surveys measured attitudes and concerns

Table 1: Privacy actions applied in the Privacy Model.

Action	Description
`Read`	The actor requests to read the value of personal data.
`Collect`	The actor measures or monitors the user in some way and then collects this personal data.
`Write`	The actor can change the value of personal data e.g. after performing `collect`.
`Disclose(actor)`	The actor discloses a particular piece of personal data to another actor.
`Store (Reference)`	Personal data is stored to a location (e.g. database, service) accessible by others.
`Delete`	The actor deletes the particular piece of data from a store.
`Redact`	The actor suppresses (or removes) the data from its disclosed form.
`Anonymize (method M)`	A particular piece of data (e.g. quasi-identifier) is anonymised using method M.
`Pseudo-anonymise (method M)`	A particular piece of data (e.g. explicit-identifier) is pseudo-anonymised using method M.

about privacy and provided data on how these attitudes and concerns change over time, particularly with the move towards a digital information age. Westin created several privacy indexes to summarize his survey results and classified the public into three broad categories:

- *High (Fundamentalist).* Fundamentalists distrust organizations that use their personal data. They choose privacy controls over consumer-service benefits when these are in conflict. About 25% of the public are privacy Fundamentalists.
- *Medium (Pragmatist).* Pragmatists weigh the benefits of opportunities against the degree of intrusiveness of personal information sought. They want the opportunity to decide whether to opt out of even non-evaluative uses of their personal information. About 57% of public fall into this category.
- *Low (Unconcerned).* The unconcerned trust organizations collecting their personal information, are ready to forego privacy claims to secure consumer-service benefits or public-order values, and are typically not in favour of the enactment of new privacy laws or regulations. About 18% of the public fall into this category.

While this research displays a clear body of evidence to suggest that users can be profiled into a particular classification based upon their answers to privacy questionnaires, it does not take into account the peculiarities of a particular user, who may differ in terms of their classification based upon the action on a particular type or piece of data. Therefore, we aim to create a user privacy profile built upon these core privacy classifications, that also captures (and predicts) the preferences for a particular service's usage of data.

One way to do this is to ask them whether they agree/disagree with each and every action by a system. This is in some sense better than the usual approach of asking users for a blanket acceptance of any action the service might take. However, for any significant system it would involve asking the user to respond to a long and detailed series of questions. For example, to accurately determine all possible results for a service with 3 active roles, 5 data categories and 3 actions, we would need to ask 45 (5*3*3) questions. No user, however interested they are in privacy, will want to go through that list, let alone answer them all accurately. To reduce the information asked from a user, we built a privacy classifier based upon a hierarchical questionnaire; this allows the privacy classifier to begin to gain an "idea" of the privacy ranking of the user. The classifier predicts the results of some (if not the majority) of the data points that we need to collect.

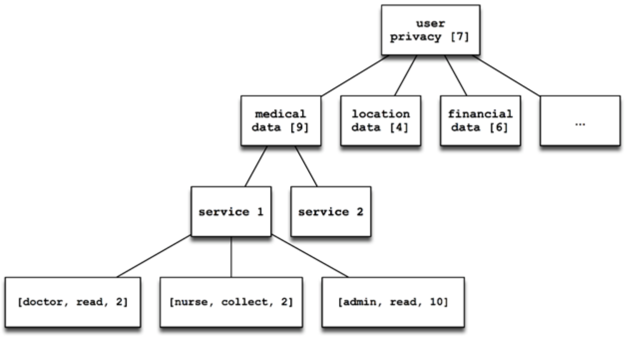


Figure 3: User Preference Model

The user preference model is therefore a set of `preferences` (captured by the questionnaire and user classification). These are illustrated and explained by the model description in Figure 3, where there are three types of preferences:

(1) The *user privacy classification* is a score between 0 and 10 that captures the user's general privacy stance. Two thresholds can be used to classify users into the three Westin types [15], the lower threshold below which they should be considered as Unconcerned, and the upper threshold at which they are considered to be a Fundamentalist.
(2) A *Data Category Privacy* is a score between 0 and 10 about a typical type of private data (e.g. medical, location and financial). For example, the user in Figure 3 is a fundamentalist about medical data. This fact is stored as `[User1, Medical Data, 9]`.
(3) A *Service action* is a score between 0 and 10 describing the user's preference for an actor to perform an action on their private data. For example, the user

in Figure 3 is unconcerned about a doctor reading medical data.

4 MODEL-DRIVEN PRIVACY ANALYSIS

We now describe the algorithm to evaluate the extent to which the behaviour of a service matches the user's privacy model, and which specific actions conflict with the user's privacy requirements. The inputs to the algorithm are as follows:

- The service behaviour model $\texttt{LTS}_{serviceA}$, which captures the behaviour of `serviceA` based on its workflows and access control restrictions.
- The User Profile $\texttt{UP}_{userid}$ is the preferences model for a user `userid`, as illustrated in Figure 3.

A model-driven analysis then combines and evaluates the models to produce the final user-centric privacy model $\texttt{LTS}_{serviceAuserid}$. In simple terms, it takes a service behaviour model and computes the user's privacy for each and every possible transition in the model.

Algorithm. Let T be the set of all transitions possible in the service. Each transition $T_{n->m}$ in T represents a transition from state S_n to S_m. These are labeled with two values to make concrete the user preferences within the LTS:

- *UserPref(Action, Data, Data Schema, Actor)*: the privacy score determined from the user preference model for the service action described in the label of the transition in terms of the Action, Data and Actor fields described earlier.
- *Risk(T_i)*: the probability of the transition T_i being executed. Where such an action exists in a service workflow this value is 1, because of course if the user decides to use the service this transition will certainly be executed. Other actions are assigned a probability based on how likely it is that the action would occur, whether by accident, by malicious design, or because unforeseen circumstances make it necessary. If someone in an admin role accessed medical data (for any reason) for 10 users out of 100, one could say $\mathrm{Risk}(\mathrm{T}_i) = 0.1$. Where data is accessed in anonymised form, the risk may be based on the strength of anonymisation, e.g. if a k-anonymised record could refer to two patients (i.e. $\mathrm{k} = 2$) then the risk of identification is 0.5, and we could assert that $\mathrm{Risk}(\mathrm{T}_i) = 0.5$.

Each value of State S_m (in $\mathrm{LTS}_{serviceA}$) is then calculated from the incoming Transition $T_{n->m}$ as follows:

$$PrivacyWeight_{nm} = Threshold_{fund}((UserPref(T_{nm}) * Risk(T_{nm})) \quad (1)$$

$$S_m = Classify(PrivacyWeight_{nm}) \quad (2)$$

`(1)` calculates a privacy score based upon the preference and risk. If the preference $User_{Pref}$ is greater than a fundamentalist score (i.e > 8) then the preference value is returned. The threshold of 8 is chosen based upon Westin's characterization of human privacy [15]. Otherwise, `UserPref * Risk(T`$_i$`)` is returned. (2) returns `agree` (where `Score` $<$ $\texttt{Unconcerned}_{Limit}$) where $\texttt{Unconcerned}_{Limit}$ is the unconcerned weighting for a user. At present, the model fixes this to 2 for all users (based upon Westin's [15] observation that 2 in 10 users are unconcerned about privacy). Otherwise `conflict` is returned. This is a simple scale assessment. For example, if the UserPref score for a transition = 7 and the risk = 0.6 then the score of 4.2 is above the threshold (`conflict`). Whereas if the UserPref score for a transition = 4 and the risk = 0.1 then the score is 0.4 and below the threshold (`agree`).

Figure 4 illustrates the algorithm applied to the running example in this paper. First, the top diagram shows the results of the first pass through the LTS model by the algorithm: each transition is labeled with the corresponding user privacy score for the action, e.g.`[collect, weight, patient record, nurse]` is replaced by `[collect, medical, nurse, 2]` from the user preference model; and `[risk, 1]` because this action comes from the expected workflow input. Each state is given the score and then this is used to compute `[agree or conflict]`. For example, a score of 2 for the first action leads to an `agree` computation. Whereas, the read medical data by the admin is in conflict because the score is higher than 8.

It is not enough to only detect conflicts on each action; it may be that an action leads to a state from which a subsequent action may occur that is in conflict. For example, we illustrate a scenario in the bottom LTS model in Figure 4. Here, `weight` has been stored in the database, but it is never subsequently used by a medical professional. However, the access control policies mean both admin and IT admin roles could read this value. These actions conflict with the user preferences, so to prevent them we make the **store** action itself in conflict, since this action creates the conditions for the subsequent potential breach of privacy. For this, when *State* S_n is in conflict the algorithm goes back to the prior *State* S_m and re-evaluates all outward transitions. For each value S_i in the set of outgoing transitions S_{out}, we reclassify S_m with the lowest value of S_{out}. Therefore, in Figure 4 the minimum value is 10 and therefore, the state must be computed as in conflict.

5 EVALUATION

To evaluate our approach we use a case-study based methodology: we apply the model-driven privacy analysis framework to a particular case study (originating from the ****** project[5]) and observe the extent to which our goals are preliminary met:

(1) That the model-driven approach accurately models user preferences and creates a model of a service's

[5] http://www.operando.eu

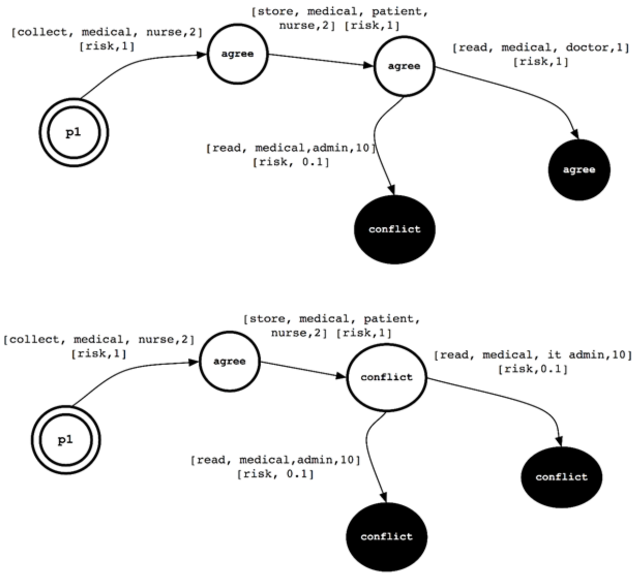


Figure 4: Privacy state computation in Service LTS models

behaviour that can be analysed with respect to an individual's privacy as opposed to a universal privacy preference. The case studies show, how the model dynamically changes for different user preferences, in order to meet this case.

(2) That the model can be used to identify privacy problems and display these implications to the user in a fine-grained manner to control their behaviour accordingly.

Here for brevity we focus on a simple case study from one of the OPERANDO project trials [2]. Ospedale San Raffaele (Milan) provides a hospital service to manage patients' diet and monitor their physical activity regimes. Patients provide nutritional data, information about their physical activity habits as well as standard health and well-being data (e.g. weight, height, BMI, etc..). Healthcare professionals are then able to monitor and control the day-to-day habits of their patients in real time.

We model data in this case study as follows. This medical service keeps two types of records about the user:

- A `Patient Record`. Which contains the fields: [name, surname, date of birth, address, health identification number, height, weight, BMI]. This record is available to actors with the roles [`doctor, dietician, nurse, or administrator`].
- A `Nutrition record`. Which contains the fields: [name, surname, health identification number, physical activity log, food intake log]. This record is available to actors with the roles [`dietician, doctor`].

The actors in the service are: i) the user, a doctor, a nurse, service administrators, and dieticians. Hence, we created an LTS to describe the initial workflow of the case study, and then given the access lists to data within the above records we generated to universal LTS for all users. A small subset of this is shown in Figure 5 (there are hundreds of states in the full model). Subsequently, we generated a set of user preferences built from five example users who answered questions regarding this online service. The execution algorithm then produced the privacy states for each user; and these are displayed in Table 2.

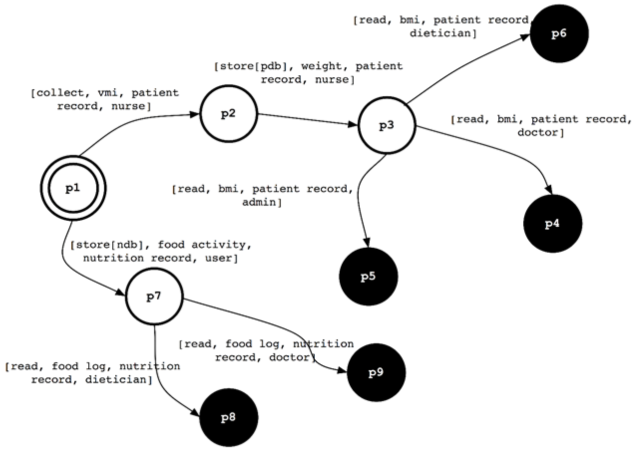

[read, bmi, patient record,
dietician]
p6
[store[pdb], weight, patient
record, nurse]
[collect, vmi, patient
record, nurse]
p2
p3
[read, bmi, patient record,
doctor]
p1
[read, bmi, patient record,
admin]
p4
[store[ndb], food activity,
nutrition record, user]
p5
p7
[read, food log, nutrition
record, doctor]
[read, food log, nutrition
record, dietician]
p9
p8

Figure 5: The Medical Service workflow LTS

These results show that the two goals are met: i) the model can accurately analyze a service's behaviour and identify privacy breaches with regard to different users with different privacy preferences; iii) the identification of the transition of concern means the privacy implication can be presented to the user (as opposed to a full model) e.g. user 4 can be notified that an administrator may be able to read their sensitive BMI field and this has been blocked based on their preferences.

6 CONCLUSIONS AND FUTURE WORK

In this paper we have argued that current approaches to privacy preservation generally take a universal approach i.e. they consider everyone to have the same level of privacy preferences. We argued that more user-centric solutions are required; however, where such solutions exist they are inherently tied to a particular system or algorithm. Therefore, to address these concerns, we presented a model-driven solution to allow users to better understand how systems use their private data. The abstract modeling allows heterogeneous systems to be controlled in order to reflect the user's privacy preferences. We demonstrated how the framework was applied to an access-controlled medical service to illustrate the effectiveness of the solution. The key contributions we have shown are: i) a formal model of user privacy within computational services and ii) a privacy aware policy computation algorithm to identify inputs to user-driven policies enforced by an access control system.

Table 2: Privacy state values for Users of the medical case study.

User	Class	P_6	P_4	P_5	P_9	P_8
1	Unconcerned	Agree	Agree	Agree	Agree	Agree
2	Pragmatist	Agree	Agree	Agree	Agree	Agree
3	Fundamentalist	Agree	Agree	Conflict	Agree	Conflict
4	Pragmatist	Agree	Agree	Conflict	Agree	Agree
5	Pragmatist	Agree	Agree	Agree	Agree	Agree

At present the models described in the paper must be initially created by a developer (in terms of the workflow) before they are automatically extended; that is, they do not have to create the complete model because the calculation of state values and unanticipated actions are computed automatically. However, an LTS may not be a natural tool for these users, so we plan to examine how we can leverage model transformation techniques to convert from a well-established standard (e.g., BPMN to model service workflows) into the privacy model needed to compute each users privacy state. The privacy models are also based upon data categories with fixed schemas e.g. weight data is medical data, name is personal data etc. This limits the extensibility of the model to personal data within this fixed world vocabulary. Therefore, we will examine the modeling of personal data using ontologies, such that reasoning logic can be applied to infer a users privacy preference for a new type of data.

ACKNOWLEDGMENTS

This work was supported by the European Commission as part of the OPERANDO project (Grant no. 653704). We also acknowledge the work of Robbie Anderson in developing an early implementation of the privacy questionnaire.

REFERENCES

[1] J Bradshaw, Andrzej Uszok, Renia Jeffers, Niranjan Suri, and others. 2003. Representation and reasoning for DAML-based policy and domain services in KAoS and Nomads. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. ACM, 835–842.

[2] Lisa Catanzaro, Luigi Clivati, and Brian Pickering. 2016. *D8.3 - Definition and planning of healthcare trials*. Technical Report. H2020 Operando Project.

[3] David W Chadwick and Kaniz Fatema. 2012. A privacy preserving authorisation system for the cloud. *J. Comput. System Sci.* 78, 5 (2012), 1359–1373.

[4] Michele Chinosi, Alberto Trombetta, and others. 2009. Integrating privacy policies into business processes. *Journal of Research and Practice in Information Technology* 41, 2 (2009), 155.

[5] L Cranor, M Langheinrich, and M Marchiori. 2002. A P3P Preference Exchange Language. (2002).

[6] Lorrie Faith Cranor. 2012. Necessary but not sufficient: Standardized mechanisms for privacy notice and choice. *J. on Telecomm. & High Tech. L.* 10 (2012), 273.

[7] Lorrie Faith Cranor, Praveen Guduru, and Manjula Arjula. 2006. User interfaces for privacy agents. *ACM Transactions on Computer-Human Interaction (TOCHI)* 13, 2 (2006), 135–178.

[8] Nicodemos Damianou, Naranker Dulay, and others. 2001. The ponder policy specification language. In *Policies for Distributed Systems and Networks*. Springer, 18–38.

[9] Roger Dingledine, Nick Mathewson, and Paul Syverson. 2004. *Tor: The second-generation onion router*. Technical Report. DTIC Document.

[10] Johnson Iyilade and Julita Vassileva. 2014. P2u: A privacy policy specification language for secondary data sharing and usage. In *Security and Privacy Workshops (SPW), 2014 IEEE*. IEEE, 18–22.

[11] Lalana Kagal, Tim Finin, and Anupam Joshi. 2003. A policy based approach to security for the semantic web. In *International Semantic Web Conference*. Springer, 402–418.

[12] Patrick Gage Kelley, Joanna Bresee, Lorrie Faith Cranor, and Robert W Reeder. 2009. A nutrition label for privacy. In *Proceedings of the 5th Symposium on Usable Privacy and Security*. ACM, 4.

[13] Pranam Kolari, Li Ding, and others. 2005. Enhancing web privacy protection through declarative policies. In *Policies for Distributed Systems and Networks, 2005. Sixth IEEE International Workshop on*. IEEE, 57–66.

[14] Tracy Ann Kosa. 2015. *Towards measuring privacy*. Ph.D. Dissertation. University of Ontario Institute of Technology.

[15] Ponnurangam Kumaraguru and Lorrie Faith Cranor. 2005. Privacy indexes: a survey of Westin's studies. (2005).

[16] Thomas Leonard, Martin Hall-May, and Michael Surridge. 2013. Modelling Access Propagation in Dynamic Systems. *ACM Transactions on Information and System Security (TISSEC)* 16, 2 (2013), 5.

[17] Yin Hua Li, Hye-Young Paik, and Boualem Benatallah. 2006. Formal consistency verification between BPEL process and privacy policy. In *Proceedings of the 2006 International Conference on Privacy, Security and Trust*. ACM, 26.

[18] Richard J Lipton and Lawrence Snyder. 1977. A linear time algorithm for deciding subject security. *Journal of the ACM (JACM)* 24, 3 (1977), 455–464.

[19] Jiajun Lu, Zhiqiu Huang, and Changbo Ke. 2014. Verification of Behavior-aware Privacy Requirements in Web Services Composition. *JSW* 9, 4 (2014), 944–951.

[20] Alessandra Mazzia, Kristen LeFevre, and Eytan Adar. 2012. The PViz comprehension tool for social network privacy settings. In *Proceedings of the Eighth Symposium on Usable Privacy and Security*. ACM, 13.

[21] Aleecia M McDonald and Lorrie Faith Cranor. 2008. The cost of reading privacy policies. *ISJLP* 4 (2008), 543.

[22] Toby Murray and Gavin Lowe. 2009. Analysing the information flow properties of object-capability patterns. In *International Workshop on Formal Aspects in Security and Trust*. Springer, 81–95.

[23] Norman Sadeh, Ro Acquisti, Travis D Breaux, and others. 2013. The usable privacy policy project. (2013).

[24] Percy Pari Salas and Padmanabhan Krishnan. 2008. Testing privacy policies using models. In *Software Engineering and Formal Methods, 2008. SEFM'08. Sixth IEEE International Conference on*. IEEE, 117–126.

[25] Stuart Short and Samuel Paul Kaluvuri. 2011. A data-centric approach for privacy-aware business process enablement. In *International IFIP Working Conference on Enterprise Interoperability*. Springer, 191–203.

[26] Fred Spiessens, Jerry den Hartog, and Sandro Etalle. 2009. *Know What You Trust*. Springer Berlin Heidelberg, Berlin, Heidelberg, 129–142. DOI:http://dx.doi.org/10.1007/978-3-642-01465-9_9

[27] Latanya Sweeney. 2002. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 05 (2002), 557–570.

[28] Yang Wang, Pedro Giovanni Leon, Kevin Scott, Xiaoxuan Chen, and others. 2013. Privacy nudges for social media: an exploratory Facebook study. In *Proceedings of the 22nd International Conference on World Wide Web*. ACM, 763–770.