Safeguard Privacy for Minimal Data Collection with Trustworthy Autonomous Agents

Mengwei Xu
Newcastle University
Newcastle upon Tyne, United Kingdom
mengwei.xu@newcastle.ac.uk

Louise A. Dennis
The University of Manchester
Manchester, United Kingdom
louise.dennis@manchester.ac.uk

Mustafa A. Mustafa
The University of Manchester & COSIC, KU Leuven
United Kingdom & Belgium
mustafa.mustafa@manchester.ac.uk

ABSTRACT

Ensuring digital privacy necessitates users giving well-considered consent to online service providers for data usage, creating an unsustainable and error-prone decision load. Software privacy agents can help make data consent decisions on behalf of users, but a compromised agent could be more detrimental than the absence of such an agent. In response, we employ trustworthy autonomous agents to safeguard users' privacy at the stage of data collection. Drawing upon General Data Protection Regulation (GDPR) principles, notably data minimisation, our autonomous agent guarantees that GDPR-reflected privacy requirements are met through strong proof. We provide a computational encoding of a typical data collection scenario—where data are requested and decisions are made about these requests—as a cognitive agent that makes decisions based on how an agent's beliefs and goals lead to particular choices. Importantly, our approach provides verifiable assurance about decisions made by these cognitive agents through formal verification, supporting both simultaneous (data requested at the same time) and sequential (data requested one after the other) situations. We provide a templated implementation of these privacy agents and a small example of a mobile app serves to illustrate how a privacy agent can be designed in practice. An in-depth evaluation is given to demonstrate its computational practicality in making privacy decisions in real time and its computational complexity in verifying them. This approach represents a promising step towards trustworthy computational stewardship in data management.

KEYWORDS

Privacy; Data Collection; Trustworthy Autonomous Agents

ACM Reference Format:
Mengwei Xu, Louise A. Dennis, and Mustafa A. Mustafa. 2024. Safeguard Privacy for Minimal Data Collection with Trustworthy Autonomous Agents. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 9 pages.

1 INTRODUCTION

In the digital era, we are surrounded by online services that require various aspects of data usage, for example, navigation systems collect location data from users for real-time traffic updates. Once the collection of users' information is required or allowed, the issue of privacy comes to the fore [5]. Users are unlikely to trust these digital service providers due to concerns about privacy breaches exemplified in the Cambridge Analytica scandal [8], where millions of Facebook users' data were exploited for political advertising.

That said, the reality is that we often trade our data for seamless access to services [9] from social media apps delivering personalized content to E-commerce websites offering targeted promotions. Though some personal data collection can be justified, service providers may collect excessive personal data that is not necessary for the intended purpose (seen in the common practice of "accept all cookies" option on websites) for business monetisation. Therefore, users need to protect their privacy and share only the necessary and minimal amount of personal information needed to access the services. This is also known as the principle of data minimisation laid out in the General Data Protection Regulation (GDPR) [31].

Predatory data collection practice exploits the complex nature of privacy in data collection in two key ways. Firstly, the growing diversity and volume of personal data requested online creates an incredible decision load on the user, constantly probing users for consent. It makes it impractical for users to identify on-the-fly if their personal data is being collected to the extent necessary, let alone discern the precise combination of data that could reveal highly sensitive personal attributes like sexual orientation or political views [2]. In fact, it often results in users completely ignoring any privacy details just to proceed. Secondly, individual users may have varying degrees of privacy expectations [23] with diverse privacy requirements for different applications. These two factors contribute to an environment where users are often prompted to accept blanket privacy policies, without comprehending the data-sharing implications or the nuances of privacy issues.

Research exists to protect the privacy of the sharing of personal data. One approach is to apply the agent-based paradigm to autonomously determine when, how and what personal data should be disclosed. At its most general, an agent is an abstract concept that represents an autonomous computational entity that makes its own decisions [40]. For example, the work [41] proposed an agent that can block access to a website or automatically notify online users if the website's privacy policies are not in line with their pre-specified privacy preferences. Similar approaches to privacy protection can also be found in [10, 11, 25, 32, 34, 35]. However, a compromised agent may pose a greater risk to privacy than the absence of such an agent [17]. Therefore, selecting agents as the means of privacy protection necessitates the need for trustworthy agents which can be proven to adhere to privacy requirements.

This work is licensed under a Creative Commons Attribution International 4.0 License.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

In this paper, we address the trustworthiness issues in agents for privacy protection in the data collection stage. We use a cognitive agent [7, 18, 30] programmed in the Beliefs-Desires-Intentions (BDI) paradigm [28, 29] where the reasons for agent choices can be inspected and analysed. BDI agents work explicitly on the intuitive concepts of beliefs, desires and intentions. Whereas (B)eliefs explicitly represent the agent's (possibly incomplete, possibly incorrect) knowledge about itself and its external environment, (D)esires (long term goals) and (I)ntentions (goals currently being pursued) are often represented implicitly by a user-provided plan library. Each plan in the plan library is a strategy that describes how, and under what conditions (based on beliefs), an agent should react to a goal (a desire). By ascribing mental qualities to computational systems, BDI agents offer an intuitive, non-technical way to design complex systems to regulate when, what, and how data can be shared with which parties (the core of the privacy definition by Alan Westin in [39]). Crucially, agents constructed in this way make it amenable to formal verification [20] to provide assurance about decisions made by agents. For example, we can verify not only what the agent decides, but what it believed that made it decide in this way, which is *unseen* in any previous work. While we promote the intuitive alignment and computational efficacy of BDI agents (which we can trust) in addressing the typical data collection challenges users encounter daily, it's worth noting that other *verifiable* agents with data-sharing decision capabilities could potentially be applicable. Nonetheless, a thorough exploration of/comparison between these alternative agents is beyond the scope here.

To design a BDI agent to safeguard data collection for users, we encode data collection requests as goals and translate human-readable privacy requirements into machine-readable BDI plans. As such, by responding to these data collection goals with BDI plans, we can inspect and analyse if the right choice has been made for these data collection requests. To conduct any meaningful analysis about any agent, we construct the environment in which the agent is situated to support scenarios of data requested at the same time (*simultaneous*) or one after the other (*sequential*) We have also drawn upon some privacy principle outlined by the GDPR and reflected them in our agent design. For example, the GDPR principle of data minimization, which mandates collecting only the minimum necessary data, is reflected computationally by maximising the condition under which a plan rule can be applied. Finally, we apply Model Checking Agent Programming Languages (MCAPL) framework [14], which provides model checking facilities to the underlying BDI agents to ensure that such an agent always makes the right decision for any data collection request.

We make the following research contributions on employing trustworthy agents to safeguard user's privacy at data collection:

- a formal description of a generic data collection problem, determining *e.g.* which personal data of users should or should not be disclosed to which data request parties;
- an agent-centric solution to the above data collection problem using the BDI paradigm together with the details of translation between the two;
- a templated implementation of a BDI agent in the model checking-focused language Gwendolen and its subsequent verification through the MCAPL framework;
- an illustration of our approach in a simple app-based scenario, verifying several properties of our privacy agent;
- an in-depth evaluation for computational practicality in simulation and scalability complexity in verification.
- discussion of our approach (pros and cons) and some future work (*e.g.* managing conflicting privacy requirements).

Outline. In Section 2 we survey the related work. In Section 3 we introduce Gwendolen (model checking-focused BDI) language. In Section 4, we propose our approach. In Section 5, we provide templated implementation of our approach and illustrate our approach in action through a simple example. In Section 6, we evaluate our approach. We discuss and conclude our approach in Section 7.

2 RELATED WORK

Privacy has long been an issue to the human race [38], usually in the form of the tendency toward territoriality that most animals have. Due to the explosive growth of information technologies, however, privacy has become a fast-growing area wherever there is a risk that people's personal data may be abused in any data-related activities. These activities include data collection, processing, and dissemination, to name a few. The modern interpretation of privacy, in particular, the implication of technology in privacy, is conceived in the seminal work [37], which argues for recognising a legal right to privacy. We mainly focus on the agents for various forms of privacy protection but will also succinctly discuss the related privacy work on each data-related activity for completeness.

Besides related work in Section 1, others have explored utility-based and ontology-based to protect privacy. For example, one latest work [24] proposed a privacy assistant that computes the trust of the situations to decide whether to make a data sharing decision on behalf of the user, or delegate the decision to the user. Though useful especially in unforeseen situations, it is, however, difficult to ensure whether the agent which computes trust is trustworthy or not. Indeed, when privacy becomes a utility-based optimisation problem, the decision-making process can rapidly become opaque and, subsequently, challenging to interpret. This issue becomes especially evident with machine learning techniques [3], resulting in low confidence in privacy-sensitive situations. For example, work [16] develops a Privacy Wizard model, which establishes personalized privacy preferences between users, utilising active learning techniques. Meanwhile, ontology-based work (reviewed in [19]) provides expressive frameworks to represent complex privacy requirements. However, these ontologies (which can be more descriptive than procedural) struggle with correctly reasoning behaviours, necessitating strong proofs for trustworthiness.

One of the closest works to us is perhaps the work [21], which proposed an approach to detect and predict the privacy agreements between users and an online social network operator. To achieve so, it uses model checking to detect if relations among the users will result in the violation of privacy agreements. Contrary to their approach of translating the privacy protection in online social networks as a model checking problem, we translate it (at the data collection stage) as an agent design issue and subsequently verify the agent using an off-the-shelf verification tool to ensure trustworthiness. We believe that our agent-centric approach offers a more natural and intuitive solution compared to a direct, expert-driven model-checking approach. Our approach is also closely related to works in addressing privacy issues in access control. For example, the work [27] looked at privacy issues for access control in the health care system. Similar to work [24], it proposed an access control model which calculates privacy rating for both the user and the data for access control. Again, it is challenging to verify these opaque utility-based techniques.

After data has been collected, with the unprecedented amount of data available, there is a large body of privacy work on machine learning for data processing. For example, works such as [1, 36] introduced the privacy-preserving concept when conducting data mining, often through anonymisation of data. Their focus is to modify or remove personal identifiers in a way that prevents the identification of individuals while maintaining the utility of the data for analysis. Instead, our aim is to hinder predatory data collection in contexts where users often feel compelled to consent to each single data collection. Once data has been collected and processed, there is a danger that data may be disseminated to other third parties without the consent of the person to whom the data relates. We acknowledge the challenge of verifying third-party compliance (with some promising work though in *e.g.* [4]), but our approach can be applied to ensure that data is shared only with certified dissemination-safe service providers. Finally, for the stage of actual data transferring and storage, it has well-established privacy protection techniques such as conventional encryption [6], protecting confidentiality in data transit/storage against unauthorised parties.

3 BACKGROUND

Gwendolen [12] is a verifiability-focused language for BDI agents with a formal operational semantics. Each Gwendolen agent moves through a *reasoning cycle* polling an external environment for perceptions and messages; converting these into a set of agent's beliefs (denoted as B) and creating *intentions* from *events*, which can be the acquisition of new beliefs or goals; and selecting an intention for consideration. If the intention has no associated plan body, then the agent seeks a plan that matches the trigger event that created the intention and places the body of this plan on a *deed stack* (deeds include actions, belief updates, and the commitment to goals) for the intention; the agent then processes the first deed, before again polling for perceptions. A Gwendolen agent also has a plan library (denoted as P) which is an ordered list of plans. Plans have guards which are evaluated using Prolog-style [26] reasoning with *reasoning rules* of the form h : −body and literals drawn from the agent's belief and goal bases with negation ∼ by failure.

Gwendolen uses standard syntactic conventions from BDI agent languages: +!g indicates the addition of a goal g; +b indicates the addition of a belief b; and −b indicates the removal of a belief b. Plans follow the pattern trigger : guard ← body;. The trigger represents the addition of a goal or a belief (beliefs may be acquired via perceptions from the environment or as a result of internal deliberation). For example, a trigger can be the addition of a *performance* goal +!g [perform] (which will be dropped after executing a related plan), or an *achievement* goal +!b [achieve] (for which the agent must continue attempting the applicable plans associated with the goal until it has acquired the belief b), and +b (indicating the addition of the belief b—possibly as a result of perception). The guard states conditions about the agent's beliefs or goals that must be true for the plan to be selected for execution; and the body is a stack of *deeds* the agent performs in order to execute the plan. These deeds typically involve the addition/deletion of goals and beliefs, and *actions*, which indicate code delegated to non-rational parts of the system, such as low-level control.

```
:name: car                                    1
:Initial Beliefs:                             2
at_speed_limit                                3
:Initial Goals:                               4
accelerate[perform]                           5
:Plans:                                       6
+!accelerate[perform]:{B at_speed_limit}      7
                     ← maintain_speed;        8
```

Listing 1: Gwendolen Cruise Control Agent Example

Listing 1 shows a simple example of a Gwendolen agent controlling a car to maintain the speed if it has reached the speed limit. Line 1 names the agent as car. A predicate at_speed_limit at line 3 indicates the agent initially believes it is at the speed limit. Line 5 specifies an initial goal to intend to accelerate. Finally, line 7 provides a plan to maintain the speed when at the speed limit.

4 TRUSTWORTHY PRIVACY SAFEGUARD AGENTS

We formalise a typical data collection scenario and address it through the agent-centric route using Gwendolen. We rely on standard notation first-order logic $\mathcal{L}$ built from the set of variables $\{x_1, \cdots, x_i\}$ ($i \in \mathbb{N}^+$), the set of predicates $\{P_1, \cdots, P_j\}$ ($j \in \mathbb{N}^+$), and logic constants and connectives $\{\top, \bot, \neg, \vee, \wedge\}$, and $\mathcal{P} \overset{\text{def}}{=} \text{Atoms}(\mathcal{L})$ is the set of finite grounded atomic formula in the underlying logic $\mathcal{L}$.

4.1 Data Collection Formalisation

The challenge in data collection is to determine which *personal data* should or should not be disclosed to which data request entity based on the user's *knowledge*. Our primary aim in this work is the set of any form of *personal* information that can be shared between one party to another. Therefore, *personal data* can be both directly observed data (*e.g.* that Bob is male) and inferred data (*e.g.* that Bob is healthy) if available before sharing. We begin with some basic terminology for data collection formalisation.

Definition 4.1. Personal data Dt^p is a set $\{dt_1, dt_2, \cdots, dt_n\}$ where each dt_i ($1 \le i \le n$) denotes a specific data element and $n \in \mathbb{N}^+$ stands for the set of natural integers.

Definition 4.2. The knowledge of the user for the collection of any personal data is defined as a knowledge base $\mathcal{K}$ such that $K \in 2^{\mathcal{P}}$ where $\mathcal{P}$ stands for the set of grounded atomic formula in a first-order logic language $\mathcal{L}$.

Example 4.3. Consider a mobile app called *vibe* that collects from users personal data $\{dt_1, dt_2, dt_3\}$ where $dt_1 = name$, $dt_2 = date\_of\_birth$, $dt_3 = email\_address$. The knowledge of a user about this mobile app can be: $\mathcal{K} = \{rate(vibe, good), verified(vibe, store), use(vibe, security\_measures\}$. This knowledge shows this app has a good rating, it is verified by the app store, and it uses appropriate security measures, *e.g.* standard encryption.

Definition 4.4. Given personal data Dt^p, the possible actions of users with respect to Dt^p are denoted as an action library $\Lambda(Dt^p) = \{P_k^1(Dt), \cdots, P_k^l(Dt) \mid Dt \subseteq Dt^p\}$ where are $\{P_k^1, \cdots, P_k^l\} \subseteq \{P_1, \cdots, P_j\}$ are predicates in the language $\mathcal{L}$, Dt is any subset of personal data which is subject to these specified action controls.

The action library contains a set of actions from the language predicates that a user can perform in principle[1]. For example, some of the most intuitive types of predicates we are likely to be interested in would be $approve(Dt)$ and $decline(Dt)$. For notational convenience, we denote the action performed on any data set as $act(Dt)$. To capture who may request data from users, we denote the possible set of these data requester entities (*e.g.* apps) as a finite set $\mathcal{N}^P$ s.t. $\mathcal{N}^P \subseteq \{x_1, \cdots, x_i\}$ $(i \in \mathbb{N}^+)$ in the language $\mathcal{L}$.

Definition 4.5. Given personal data Dt^p and the set of data requester entities $\mathcal{N}^P$, a privacy requirement is defined as a 4-ary tuple $\langle \mathcal{N}, \phi, Dt^{dis}, act(Dt)\rangle$ where $\mathcal{N} \subseteq \mathcal{N}^P$ is a set of data requester entities variables in the language $\mathcal{L}$, ϕ is any arbitrary formula in $\mathcal{L}$, Dt^{dis} stands for the disassociation data set s.t. $Dt^{dis} \subseteq Dt^p$, and $act(Dt)$ denotes an action control predicate in $\Lambda(Dt^p)$ to respond to the collection of any subset of personal data $Dt \subseteq Dt^p$.

Each privacy requirement defines the specific circumstance under which what data control action should be performed on any given subset of personal data $Dt \subseteq Dt^p$ based on who requests it. The variable set $\mathcal{N}$ registers the list of entity names that can access the data in some form. The formula ϕ denotes the condition that must be satisfied to perform the action $act(Dt)$ and Dt^{dis} stands for the disassociation data set which cannot be requested together with Dt (*i.e.* the combination of them can lead to derivative knowledge about users). Whereas the condition ϕ limits the collection of personal data to only what is necessary for a specific purpose, the disassociation data set Dt^{dis} prevent data linkability. It is particularly useful to have disassociation data set specified when the relevant action is to decline sharing requested data, which we will see in detail later on in Section 5.2.

Example 4.6. Consider the same app in Example 4.3, that also collects specific data element $dt_4 = monthly\_income$ and $dt_5 = daily\_expenses$. However, knowing a user's *monthly_income* and *daily_expenses* provides insights into their finances (*e.g.* financial affordability). Combining them with user's name opens the possibility of *e.g.* financial fraud. To prevent it, we formalise the following privacy requirement for users where $\top$ denotes the truth value.

$$\langle \{vibe\}, \top, \{name, monthly\_income\}, decline(\{daily\_expenses\})\rangle$$

This requirement specifies the disassociation relationship between $\{name, monthly\_income\}$ and $\{daily\_expenses\}$. And users should decline the request of $\{daily\_expenses\}$ from the app *vibe* when it believes $\{name, monthly\_income\}$ is also under request.

Definition 4.7. Given personal data Dt^p and the set of data requester entities $\mathcal{N}^P$, a function $\chi : 2^{Dt^p} \times 2^{\mathcal{N}^P} \to \mathcal{L}$ is called request function where $\mathcal{L}$ stands for the underlying logic language.

[1]Unfortunately, there are a variety of deliberate strategies from service providers that circumvent the intent of privacy by design [33], *e.g.* by limiting these choices of users, which we believe is crucial to demonstrate respect for user privacy.

The function χ records the request of any data set by data requesters as a logic formula in $\mathcal{L}$. As such, we can reason with the concurrent data request from different parties, such as some combination of data sets that may pose a privacy risk.

Definition 4.8. A privacy requirement $\langle \mathcal{N}, \phi, Dt^{dis}, act(Dt)\rangle$ is *relevant* with respect to a knowledge base $\mathcal{K}$ iff $\mathcal{K} \models \chi(Dt, \mathcal{N})$ (*i.e.* personal data Dt is believed to be under request by entities $\mathcal{N}$).

Definition 4.9. A privacy requirement $\langle \mathcal{N}, \phi, Dt^{dis}, act(Dt)\rangle$ is *applicable* with respect to a knowledge base $\mathcal{K}$ iff (i) it is relevant and (ii) $\mathcal{K} \models \phi \wedge \chi(Dt^{dis}, \mathcal{N})$ where $\chi(\emptyset, \mathcal{N}) \overset{\text{def}}{=} \top$ (*i.e.* the request of empty disassociation data set is always true).

Definition 4.8 and Definition 4.9 are the core of our data collection formalism. Definition 4.8 says that a privacy requirement comes into relevance only when personal data, which is subject to specified action control, is recorded as requested by a set of entities in the language $\mathcal{L}$. To be an applicable privacy requirement, two additional conditions need to be satisfied. The first is the necessity of condition ϕ being satisfied. The second is dependent on the existence of disassociation data set; if present, such data set must be under request and be entailed (*i.e.* $\mathcal{K} \models \chi(Dt^{dis}, \mathcal{N})$).

It's important to highlight here that Definition 4.9 inherently prioritises the declination of data requests in situations where disassociation data is present. This focus on declination implicitly places privacy preservation at a higher priority than data accessibility. This distinctive feature of our framework takes an proactive approach to respecting and upholding user privacy and, as we'll demonstrate in later sections through formal verification, it ensure data request declination will be respected whenever needed.

We close the section by noting that what we have done so far is to define the relevant concepts of data collection at the abstract and general levels. The subject of the following section is its computational encoding in Gwendolen agents.

4.2 Agent Design

In this section, we encode a data collection problem formalised in Section 4.1 as a BDI agent in Gwendolen language. Recall our goal is to design a privacy agent that 1) can autonomously determine when, how and what personal data should be disclosed to whom and 2) is proven to be trustworthy. Regarding trustworthiness, our main goal is to prove that the agent always endeavours to act in line with privacy requirements and never deliberately chooses options that it believes it should not choose.

Before we start, to conduct any meaningful analysis about any agent, we need to consider the environment where the agent is situated, particularly the sequences of environment inputs when the agent is operating. This has considerable implications in data collection scenarios. For example, the service providers may ask the users to provide certain data and then later in time be asked to provide other data (*i.e. sequential* data request), which when combined, can raise some privacy concerns. Alternatively, personal data can be requested at the same time (*i.e. simultaneous* data request). We formalise the environment as follows:

Definition 4.10. Given personal data Dt^p and the set of data requester entities $\mathcal{N}^P$, χ the request function, and $\mathcal{K}$ the user's knowledge, an environment is defined as a set $\mathcal{E}$ such that $\mathcal{E} \in 2^{\theta}$ where we have $\theta \overset{\text{def}}{=} \mathcal{K} \cup \bigcup_{Dt \subseteq Dt^{p}, \mathcal{N} \subseteq \mathcal{N}^{p}} \chi(Dt, \mathcal{N})$.

An environment is any possible combination of (1) the request status of any possible personal data by any listed entity and (2) any subset of the knowledge of the user. As such, we can have minimal assumptions *i.e.* the environment inputs that the agent receives are on an entirely random basis. Where the random inputs of the request of personal data capture different personal data may be requested sequentially or simultaneously by different entities, the random input of the user's knowledge can account for the lost or forgotten messages from the user's end. When model checking, the random behaviour of the verification environment causes the search tree to branch and the model checker to explore all environmental possibilities to analyse the agent's behaviours for correctness.

The agent will then perceive[2] the current environment, which becomes the agent's beliefs, formalised as follows:

Definition 4.11. Given a current environment $\mathcal{E}$, we have an agent's belief set B in GWENDOLEN such that $\mathcal{E} \subseteq B$.

Definition 4.11 shows the current environment is a subset of the agent's beliefs as the agent may make some internal mental notes or bookkeeping, which we will utilise in the next definition. We note it is the topic of Section 5 providing technical solutions on how to pass these environmental inputs to the agent in GWENDOLEN.

To actually make these data-sharing decisions, the agent will need to check against the set of privacy requirements. Recall that a plan in GWENDOLEN has the form trigger : guard ← body; where trigger represents the addition of a goal/belief, guard the context condition, and body the plan-body. We have the following computational encoding of privacy requirements as plans in GWENDOLEN.

Definition 4.12. Let $\langle \mathcal{N}, \phi, Dt^{dis}, act(Dt) \rangle$ be a privacy requirement. We can represent it a GWENDOLEN plan as follows:

$$+!pr(Dt, \mathcal{N})[\text{perform}] : \{\text{B}\phi, \text{B}\chi(Dt^{dis}, \mathcal{N})\} \leftarrow +act(Dt), act(Dt).$$

The translation in Definition 4.12 from privacy requirements to GWENDOLEN plans is crucial. Here, the task of checking privacy requirements on personal data Dt requested by $\mathcal{N}$ becomes the *trigger* (symbolically denoted as $+!pr(Dt, \mathcal{N})$ [perform]) of the plan, and it is a performance goal. To respond to a performance goal, the agent needs to search for a plan for it, execute the plan if applicable and then drop the goal when completed. The guard of the plan is the conjunction of the condition ϕ, and the request status of the disassociation data set $\chi(Dt^{dis}, \mathcal{N})$. Finally, the body of the plan consists of the bookkeeping of the execution of this action through the addition of a belief atom (*i.e.* $+act(Dt)$) and the actual specified action control on the requested personal data *i.e.* $act(Dt)$. We hightlight that the addition of the belief bookkeeping can allow us to inspect the agent easily for correct decisions.

Example 4.13. Continuing Example 4.6, we have a translation from a privacy requirement (top) to a GWENDOLEN plans (bottom):

$$\langle \{vibe\}, \top, \{name, monthly\_income\}, decline(\{daily\_expenses\}) \rangle$$

$$\begin{gathered} +!\text{pr}(\{\text{daily\_expenses}\}, \{\text{vibe}\})[\text{perform}] : \\ \text{B}\ \chi(\{\text{name}, \text{monthly\_income}\}, \{\text{vibe}\}) \leftarrow \\ +\text{decline}(\text{daily\_expenses}), \text{decline}(\text{daily\_expenses}) \end{gathered}$$

[2]The partial observability in model checking BDI agents remains arguably unsolved and it is not supported in MCAPL either.

In the GWENDOLEN plan above, the guard checks if the name and monthly income are requested by the app called *vibe*, and if so, the action is to decline daily expenses to preserve privacy.

4.3 Verifying Agent

GWENDOLEN agents can be verified using the Agent JavaPathFinder (AJPF) model-checker that is available with GWENDOLEN as part of the MCAPL [13]. To prove our privacy agent always acts in line with given privacy requirements and never deliberately acts in a way that it believes will violate these requirements, we use AJPF's property specification language, which is based on propositional linear-time temporal logic (LTL) [15]. This specification language uses (among other things) modalities to describe an agent: $\text{B}(ag, f)$ means that agent ag believes formula f. The language also has standard constructs for LTL *e.g.* conjunction &, disjunction ||, negation ~, implication ->, eventually <>, and always [].

Example 4.14. Continuing Example 4.13, we can automatically check that it is always the case that if the agent believes that the name and monthly income are under request, it will eventually believe that it declines the request for daily expenses. Such a property can be formalised as follows:

```
[](B(agent, χ({name, monthly_income}, {vibe}))
    -> <> B(agent, decline(daily_expenses)))
```

The property above shows a typical behaviour that we check. The actual agent's behaviours often require a conjunction of some extended properties of this kind to hold. For example, we may check that "if the agent believes the name and monthly income are under request and it has not made any decision on them yet, then it will believe that it eventually declines the request for daily expenses".

5 IMPLEMENTATION AND EXAMPLES

We provide an agent-centric implementation of the data collection problem defined in Section 4 in the MCAPL framework. This implementation is illustrated in Fig. 1 where we have an environment and privacy agent itself. The environment, which is external to our privacy agent, is an abstraction of any entity (who may request personal data from the users) and users themselves (who have their current knowledge of the situations). This information of the request of personal data (*not the requested data itself*) and the user's knowledge in the environment can be passed to the privacy agent through means such as messages. The privacy agent with privacy requirements will then decide on each request and recommend the appropriate decisions. We also note that the environment here does not model the actual decision that the user eventually makes as modelling human behaviour is a complex task. It can be difficult (and out of the scope here) to accurately predict how a person will behave in a given situation despite specific instructions [22].

5.1 Agent Design Template

To design an agent system illustrated in Fig. 1, A GWENDOLEN template for our privacy agent is given in Listing 2.

```
:name: privacy_agent
:Plans:
+.received(:tell,B):{True} ← +B;
+request(D,N):{True} ← +!pr(D,N)[perform];
```

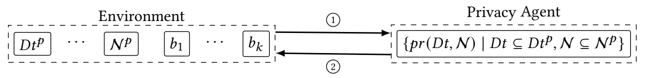
Environment
Dt^p ⋯ $\mathcal{N}^p$ b_1 ⋯ b_k
①
②
Privacy Agent
$\{pr(Dt, \mathcal{N}) \mid Dt \subseteq Dt^p, \mathcal{N} \subseteq \mathcal{N}^p\}$

Figure 1: Environment has personal data Dt^p that may be requested by the list of entities $\mathcal{N}^p$ and the knowledge of the user $\mathcal{K} \overset{\text{def}}{=} \{b_1, \cdots, b_k\}$ (①). The privacy agent provides recommendations (②) as per the set of possible privacy requirements $\{pr(Dt, \mathcal{N}) \mid Dt \subseteq Dt^p, \mathcal{N} \subseteq \mathcal{N}^p\}$.

```
+!pr(Dt_1, N_1)[perform]:{guard_1} ←                          5
   +act(Dt_1),.send(user,:perform,act(Dt_1)),;                6
...                                                           7
+!pr(Dt_j, N_j)[perform]:{guard_j} ←                          8
   +act(Dt_j),.send(user,:perform,act(Dt_j));                 9
```

Listing 2: Gwendolen template for privacy agent in Fig. 1

A quick commentary of our privacy agent is as follows. The environment will send the privacy agent the messages that trigger its operation such as who is requesting which data *i.e.* request(D, N). In Gwendolen, when a message is received, it is turned in an event .received(: tell, B). Hence, we have a plan on line 3 to address this event to receive any request for personal data together with the user's knowledge from the environment and turn them into its beliefs. The addition of data requests (+request(D, N)) will result in the addition of the task to check relevant privacy requirements (line 4). Finally, plans on lines 5-9 are the encoded Gwendolen plans for the privacy requirements where the formulation of the guard is subject to Definition 4.12. Instead of sending users the recommended actions as beliefs, the agent sends them as goals for the users to adopt (*i.e.* .send(user, : perform, act(Dt))—an internal action supported in MCAPL), which can potentially be passed through some APIs to users in the form of a user interface. Since our focus is the design of the privacy agent in this work, we do not touch the development of these potential user interfaces.

5.2 Health App

We look at a simple example of a mobile app for health management with primary intention on illustrating how to write these privacy agents in practice. The extensive computational evaluation of our approach will instead be given in Section 6. Users can download a mobile app that claims to help them monitor their daily physical activity, such as steps taken and calories burned. The app also offers personalised workout plans and diet recommendations based on your input. The app may require you to provide extensive personal information at the registration stage or during the usage stage. This personal information can (1) be irrelevant to the purpose of this app or (2) lead to derived knowledge of the users that can lead to serious privacy and security issues.

Listing 3 gives the set of messages through which this kind of mobile app, namely as *vibe* again, may request and the knowledge of a typical user. These messages request personal data[3] (lines 1-6), including information of the users on full name, date of birth (dob), gender, body mass index (bmi), contact, home address, marital status, occupation, education, and GPS (gps). The user's knowledge (lines 7-9) shows that the user is aware that it has received a workout plan and diet recommendation from the app, and this app monitors its daily physical activities.

```
request(full_name,vibe),request(dob,vibe),                    1
request(gender,vibe),request(bmi,vibe),                       2
request(contact,vibe),request(home_address,vibe),             3
request(marital_status,vibe),                                 4
request(occupation,vibe),                                     5
request(education,vibe),request(GPS,vibe)                     6
belief(workout_plan(present)),                                7
belief(diet_recommendation(present)),                         8
belief(daily_physical_activities(monitored))                  9
```

Listing 3: A set of potential environment inputs

Since this health app can, in principle, request any of the data in Listing 3 at any time and the user may or may not share all knowledge with the privacy agent at one go, the environment we construct will have to be able to assert these data requests and user's knowledge on an entirely random basis as environmental inputs that the agent receives. The MCAPL framework provides support for creating these kinds of environments for Gwendolen programs through a dedicated class that can, in turn, be sub-classed. The sub-classes simply have to sub-class the methods for generating random messages *i.e. generate_messages* for our possible set of messages in our case. When model checking, the random behaviour of the environment leads to an analysis of the agent's behaviours under all possible environmental possibilities.

We now analyse the potential privacy risk if any sub-list of data were requested by this health app. First of all, the app requests personal information that seems unrelated to its core functionality. For example, the user's marital status, occupation, and education level should not impact the app's ability to track their health progress or provide workout plans. As such, any of them should be rejected. For instance, the privacy requirement for data of marital status is given in Listing 4 where the other two can be similarly given.

```
+!pr(marital_status,vibe)[perform]:{True} ←
   +decline(marital_status),
   .send(user,:perform,decline(marital_status));
```

Listing 4: Data collection decline plan

Secondly, the excessive and seemingly irrelevant personal information collected by this app can lead to derived knowledge about users, potentially resulting in serious issues, including *identity theft*, *security risks*, and *discrimination.* We now detail these issues one by one and, importantly, propose some suitable privacy requirements to mitigate these issues, though our suggestions are by no means exhaustive. But it does demonstrate the flexibility and generality of our approach due to the expressive nature of symbolic agents.

Identity Theft. With the full name, date of birth, gender, home address, and contact, a malicious actor could impersonate the user, apply for loans or credit cards, and potentially gain unauthorised access to the user's financial accounts. To avoid this, a sensible solution would be to provide the data of age_range (which is sufficient for health analysis) instead of data of dob through the plan in Listing 5. We highlight that due to the underlying first-order logic, it enables us to construct expressive predicates for data action

[3]In this simple case, each requested personal data is a singleton, hence no set symbol notation for clarity. The same applies to the entity requesting data.

control. By allowing users to negotiate alternative data, giving them more control over their personal information. As such, users can provide a less revealing version of the requested data, *e.g.* limiting sensitive information while still accessing services.

```
+!pr(dob, vibe)[perform]:{True} ←
   +substitute(dob, age_range),
   send(user, :perform, substitute(dob, age_range));
```

Listing 5: Data collection substitution plan

Home Security Risks. While both home address and GPS data pose privacy risks individually, the risk to privacy arises significantly when home address data is used in combination with GPS data. The home address reveals 'where' you live, but GPS data can reveal 'when' you are away. When combined, these pieces of information allow for more precise targeting of crimes such as burglaries. To prevent it, these two data can be disassociated given in Listing 6.

```
+!pr(GPS, vibe)[perform]:{
 B request(home_address, vibe)} ←
 +decline(GPS),.send(user, :perform, decline(gps));
```

Listing 6: Data collection decline plan for GPS

Health Insurance Discrimination. By analysing the user's daily physical activities, workout plans, dietary recommendations, date of birth, gender, and BMI, the app could infer potential health risks or medical conditions. This sensitive information could be shared with insurance companies, which could result in higher premiums or even denial of coverage. A quick way to avoid this is to disassociate the data of bmi from the rest of these data shown in Listing 7.

```
+!pr(bmi, vibe)[perform]:{
 B request(dob, vibe), B request(gender, vibe),
 B workout_plan(present),
 B diet_recommendation(present),
 B daily_physical_activities(monitored) } ←
 +decline(bmi),.send(user, :perform, decline(bmi));
```

Listing 7: Data collection decline plan for BMI

Finally, we have a plan in Listing 8 to approve other personal data that are not already subject to any specified action control. In Gwendolen, plans listed first are given higher priority. The plan in Listing 8, which is put at the list's end, will only execute if no preceding plans, such as data declination, apply.

```
+!pr(D,N)[perform]:{True} ←
     +approve(D),.send(user, :perform, approve(D));
```

Listing 8: Data collection approve plan

Verification Capability. We have verified standard safety and liveness properties. For example, if the set of action controls available includes "approve, decline, substitute" data requests. Then, the safety properties ensure that the privacy agent will neither approve, decline, nor substitute a data request all at once (safety) and that if data were requested, there would always be some response eventually (liveness). Some of these formalisation are given as follows where _ denotes a wildcard which is supported natively by MCAPL:

safety property:
```
[](~ B(ag, approve(gps)) || ~ B(ag, decline(gps)) || ~
        B(ag, substitue(gps, _)))
```
liveness property:
```
[](B(ag, request(gps, vibe)) → <>(B(ag, approve(gps))
   || B(ag, decline(gps)) || B(ag, substitute(gps, _))))
```

We also proved properties specific to the user's privacy requirements. For example, if the privacy agent believes that the data of home address B(ag, request(gps, vibe)) and GPS are under request by the app *vibe* and the agent has not either approved GPS or substituted it with something else, the data of GPS will be declined eventually. This property can be formalised as follows:

```
[](B(ag, request(home_address, vibe)) & B(ag, request(gps, vibe))
   & ~ B(ag, approve(gps)) & ~ B(ag, substitute(gps, _)) →
            <> B(ag, decline(gps)))
```

The implementation and the full list of proven properties can be found in the GitHub distribution[4].

6 EVALUATION

We provide an in-depth evaluation of our approach both 1) in simulations to demonstrate its capability for run-time privacy decision-making and 2) in verification to show the computational complexity to guarantee the correctness of the design of such agents. Here by *simulation*, we mean simply running an agent and *verification* means analysing an agent. We are particularly interested in simultaneous data requests for simulation evaluation to ensure our approach can indeed relieve humans with this overwhelming data request in an acceptable time. Meanwhile, we shed insights into how expensive verification can be and how the computational expensiveness varies in different settings. All results are obtained on a laptop with a 16-core Intel Core i7-11800H at 2.30GHz (hyperthreaded), 16 GB memory, and running 64-bit Ubuntu Linux 20.04.3 LTS. Again, the full source codes can be found in the previous GitHub link.

6.1 Evaluation Setting

In the interests of generality, our evaluation is based on a set of randomly generated, synthetic parameterised data requests and relevant Gwendolen plans representing the privacy requirements to address these data . By varying the number of data which may be requested and plans for these data requests, and whether data is requested in a simultaneous (*i.e.* data requested at the same time) or sequential (*i.e.* data requested one after the other) manner, we can evaluate the performance of our approach. For simplicity, we assume an environment that only accounts for data requests.

6.2 Simulation

To cope with the high volume of data a user may have to consider giving consent to share in an acceptable time, we *stress test* out our approach for handling data requests in a simultaneous manner. The results are shown in Fig. 2. The left plot in Figure 2 which varies the number of data requests but assumes one plan for each data request shows its near-linear[5] performance regarding the number of data requests. For example, it took 3 seconds to respond to 100 data but around 4 minutes to respond to 5000 data. Meanwhile, the

[4]https://github.com/Mengwei-Xu/privacy_agents_mcapl
[5]The coefficient of determination for the linear regression analysis is around 0.918.

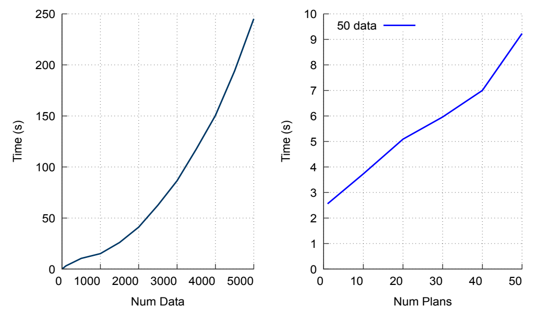

Figure 2: Simulation time increases near-linearly with the number of data and plans for simultaneous data requests.

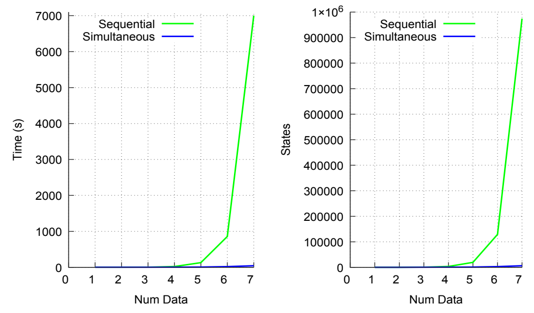

Figure 3: Transition system construction time and states increase exponentially with the number of data.

right plot in Figure 2 shows the near-linear increase of time when the number of plans increases in the case of 50 data requests. This is because the agent may need to search each plan one by one to find an applicable plan to address a given data request.

6.3 Verification

Unlike the simulation, verification is highly expensive due to its exhaustive nature in general. A crucial contribution of our evaluation is to be able to provide insights on the contrasting levels of difficulties depending on how data is requested and what level of confidence we are seeking. For example, Fig. 3 shows that it is significantly more expensive to verify our privacy agent in the sequential setting than the simultaneous one. It should come unsurprised as in the sequential data requests, we are verifying all possible subsets of data that have not been requested at all possible steps. On the contrary, in the simultaneous data requests, the complexity mainly comes at the beginning where all possible subsets of data may be requested. However, though significantly cheaper to verify than the sequential data requests, the simultaneous data requests are still exponential in their nature shown in Fig. 4.

7 DISCUSSION AND CONCLUSION

Our approach casts data collection at the user's end as a decision-making problem and employs autonomous agents to bear this responsibility. While this delegation may enhance efficiency, the trustworthiness of these agents becomes a crucial concern. Therefore, formal verification becomes indispensable to establish the confidence of these autonomous agents. In contrast to most of the previous work, we can guarantee the correct agent behaviours to handle highly sensitive personal data at the stage of data collection.

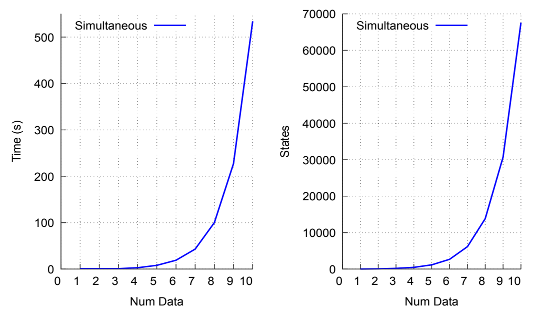

Figure 4: Transition system construction time and states increase exponentially with the number of data.

Nevertheless, our approach comes with some limitations. We note that the agent verification in the sequential setting will quickly and inevitably cause a state explosion. In practice, since verification is done at the design time, it will not cause any issues in the actual operation (shown in the simulation). That said, this issue can be mitigated by focusing on a small set of environmental inputs that will affect one another. Another issue we have not considered is the potentially conflicting privacy requirements. To address this, we have some ideas to adopt a priority-based conflict-solution framework. If it is bound to violate two privacy requirements, the agent should violate the one which will cause less damage. And Gwendolen naturally supports order-based plan selection. Finally, our framework does face another inevitable privacy threat: the possibility of an unauthorised intrusion into our privacy agent, potentially exposing user privacy requirements. Employing encryption protocols to mitigate this is a future consideration.

Overall, our approach is abstract and generic in nature, supporting both simultaneous and sequential data requests, and we have supplied a computational instantiation of the framework for proof-of-concept through autonomous agents together with some in-depth evaluation of its run-time decision practicality and expensive verification complexity. The use of autonomous systems not only highlights their capacity to tackle societal issues but also emphasises the need for user trust. Our approach showcases the intuitive alignment and effectiveness of a symbolic cognitive agent (which we can trust through strong proof) in addressing the typical data collection challenges users encounter daily.

ACKNOWLEDGMENTS

This work is supported by the University of Manchester and Newcastle University, and EPSRC, under projects Computational Agent Responsibility (EP/W01081X), UKRI Trustworthy Autonomous Systems Node in Verifiability (EP/V026801/2), and EnnCore: End-to-End Conceptual Guarding of Neural Architectures (EP/T026995/1).

REFERENCES

[1] Rakesh Agrawal and Ramakrishnan Srikant. 2000. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 439–450.

[2] Khaled Gubran Al-Hashedi and Pritheega Magalingam. 2021. Financial fraud detection applying data mining techniques: A comprehensive review from 2009 to 2019. *Computer Science Review* 40 (2021), 100402.

[3] Rob Ashmore, Radu Calinescu, and Colin Paterson. 2021. Assuring the machine learning lifecycle: Desiderata, methods, and challenges. *ACM Computing Surveys (CSUR)* 54, 5 (2021), 1–39.

[4] Masoud Barati, Gagangeet Singh Aujla, Jose Tomas Llanos, Kwabena Adu Duodu, Omer F Rana, Madeline Carr, and Rajiv Ranjan. 2021. Privacy-aware cloud auditing for GDPR compliance verification in online healthcare. *IEEE Transactions on Industrial Informatics* 18, 7 (2021), 4808–4819.

[5] Lemi Baruh, Ekin Secinti, and Zeynep Cemalcilar. 2017. Online privacy concerns and privacy management: A meta-analytical review. *Journal of Communication* 67, 1 (2017), 26–53.

[6] Rajdeep Bhanot and Rahul Hans. 2015. A review and comparative analysis of various encryption algorithms. *International Journal of Security and Its Applications* 9, 4 (2015), 289–306.

[7] Michael Bratman. 1987. Intention, plans, and practical reason. (1987).

[8] Carole Cadwalladr and Emma Graham-Harrison. 2018. Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach. *The guardian* 17, 1 (2018), 22.

[9] Nick Couldry and Ulises A Mejias. 2019. The costs of connection. In *The Costs of Connection*. Stanford University Press.

[10] Ludivine Crépin, Yves Demazeau, Olivier Boissier, and François Jacquenet. 2009. Sensitive data transaction in hippocratic multi-agent systems. In *Engineering Societies in the Agents World IX: 9th International Workshop*. Springer, 85–101.

[11] Anupam Das, Martin Degeling, Daniel Smullen, and Norman Sadeh. 2018. Personalized privacy assistants for the internet of things: Providing users with notice and choice. *IEEE Pervasive Computing* 17, 3 (2018), 35–46.

[12] Louise A Dennis. 2017. Gwendolen semantics: 2017. (2017).

[13] Louise A Dennis. 2018. The MCAPL Framework including the Agent Infrastructure Layer and Agent Java Pathfinder. *The Journal of Open Source Software* 3, 24 (2018).

[14] Louise A Dennis, Michael Fisher, Matthew P Webster, and Rafael H Bordini. 2012. Model checking agent programming languages. *Automated software engineering* 19 (2012), 5–63.

[15] E Allen Emerson. 1990. Temporal and modal logic. In *Formal Models and Semantics*. Elsevier, 995–1072.

[16] Lujun Fang and Kristen LeFevre. 2010. Privacy wizards for social networking sites. In *Proceedings of the 19th international conference on World wide web*. 351–360.

[17] Michael R. Galbreth and Mikhael Shor. 2010. The Impact of Malicious Agents on the Enterprise Software Industry. *MIS Quarterly* 34, 3 (2010), 595–612.

[18] Michael P Georgeff and A Rao. 1992. An abstract architecture for rational agents. In *Proc. of the Third International Conference on Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers Inc San Francisco, CA, USA, 439–449.

[19] Mohamad Gharib, Paolo Giorgini, and John Mylopoulos. 2020. An ontology for privacy requirements via a systematic literature review. *Journal on Data Semantics* 9 (2020), 123–149.

[20] Osman Hasan and Sofiene Tahar. 2015. Formal verification methods. In *Encyclopedia of Information Science and Technology, Third Edition*. IGI Global, 7162–7170.

[21] Özgür Kafalı, Akın Günay, and Pınar Yolum. 2014. Detecting and predicting privacy violations in online social networks. *Distributed and Parallel Databases* 32, 1 (2014), 161–190.

[22] Daniel Kahneman, Stewart Paul Slovic, Paul Slovic, and Amos Tversky. 1982. *Judgment under uncertainty: Heuristics and biases*. Cambridge university press.

[23] Nadin Kökciyan, Nefise Yaglikci, and Pinar Yolum. 2017. An argumentation approach for resolving privacy disputes in online social networks. *ACM Transactions on Internet Technology (TOIT)* 17, 3 (2017), 1–22.

[24] Nadin Kökciyan, Pınar Yolum, et al. 2022. Taking situation-based privacy decisions: Privacy assistants working with humans. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. 703–709.

[25] A Can Kurtan and Pınar Yolum. 2021. Assisting humans in privacy management: an agent-based approach. *Autonomous Agents and Multi-Agent Systems* 35 (2021), 1–33.

[26] Pierre M Nugues. 2006. *An introduction to prolog*. Springer.

[27] P Blessed Prince and SP Jeno Lovesum. 2020. Privacy enforced access control model for secured data handling in cloud-based pervasive health care system. *SN Computer Science* 1, 5 (2020), 239.

[28] Anand S Rao. 2005. AgentSpeak (L): BDI agents speak out in a logical computable language. In *7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*. Springer, 42–55.

[29] Anand S Rao and Michael P Georgeff. 1997. Modeling rational agents within a BDI-architecture. *Readings in agents* (1997), 317–328.

[30] Anand S Rao and Michael Wooldridge. 1999. *Foundations of rational agency*. Springer.

[31] Protection Regulation. 2018. General data protection regulation. *Intouch* 25 (2018), 1–5.

[32] Ralph Schäfermeier, Theodoros Mitsikas, and Adrian Paschke. 2022. Modeling a GDPR Compliant Data Wallet Application in Prova and AspectOWL. *on AI Compliance Mechanism (WAICOM 2022)* (2022), 50.

[33] Than Htut Soe, Oda Elise Nordberg, Frode Guribye, and Marija Slavkovik. 2020. Circumvention by design-dark patterns in cookie consent for online news outlets. In *Proceedings of the 11th nordic conference on human-computer interaction: Shaping experiences, shaping society*. 1–12.

[34] Monica Tentori, Jesus Favela, and Marcela D Rodriguez. 2006. Privacy-aware autonomous agents for pervasive healthcare. *IEEE Intelligent Systems* 21, 6 (2006), 55–62.

[35] Onuralp Ulusoy and Pinar Yolum. 2021. Panola: A personal assistant for supporting users in preserving privacy. *ACM Transactions on Internet Technology (TOIT)* 22, 1 (2021), 1–32.

[36] Jaideep Vaidya and Chris Clifton. 2004. Privacy-preserving data mining: Why, how, and when. *IEEE Security & Privacy* 2, 6 (2004), 19–27.

[37] Samuel Warren and Louis Brandeis. 1989. The right to privacy. In *Killing the Messenger*. Columbia University Press, 1–21.

[38] Alan Westin. 1984. The origins of modern claims to privacy. (1984).

[39] Alan F Westin. 1968. Privacy and freedom. *Washington and Lee Law Review* 25, 1 (1968), 166.

[40] Michael Wooldridge. 2009. *An introduction to multiagent systems*. John wiley & sons.

[41] Ni Zhang and Chris Todd. 2006. A privacy agent in context-aware ubiquitous computing environments. In *10th IFIP TC-6 TC-11 International Conference*. Springer, 196–205.