# WIP: Towards Privacy Compliance by Design in the Matter Protocol

Yichen Liu*, Jingwen Yan†, Song Liao‡, Long Cheng† and Luyi Xing*
*Indiana University Bloomington
†Clemson University
‡Texas Tech University
*{liuyic, luyixing}@iu.edu, †{jingwey, lcheng2}@clemson.edu, ‡song.liao@ttu.edu

***Abstract*—Privacy compliance has become a significant concern for IoT users as the popularity of diverse IoT devices continues to grow. However, the heterogeneous nature of IoT brings challenges in designing effective privacy-preserving mechanisms. While Matter is a promising unifying connectivity protocol for IoT, it currently offers limited privacy compliance features. In this position paper, we propose the MATTERCOMPLIANCE framework, which achieves privacy compliance by design within the Matter protocol. The design of MATTERCOMPLIANCE follows three principles: providing reliable and proactive privacy disclosure for users, offering interfaces for developers to conveniently integrate privacy mechanisms, and enabling users to manage their privacy settings. By integrating privacy-preserving capabilities in the Matter protocol, MATTERCOMPLIANCE fills the gap in offering a unified solution for privacy compliance in IoT systems.**

## I. INTRODUCTION

Privacy violations and compliance challenges have raised significant concerns among regulators and users. Many regulations and policies have been made to address the widely rising privacy concerns, such as the European General Data Protection Regulation (GDPR) [1] and the California Consumer Privacy Act (CCPA) [2]. The regulators define users' privacy rights such as the right to view current data collection, the right to be forgotten, and the right to control the data collection process.

Internet of Things (IoT) is a field raising privacy compliance concerns, as the devices are equipped with a bunch of sensors to detect and collect information from the surrounding environment and the users (*e.g.*, user weight, motion, environment temperature). Unauthorized information exposure from IoT devices through network traffic can lead to privacy breaches [3], [4]. Besides, recently several works propose methods to detect privacy incompliance by analyzing IoT companion apps [5], [6] or devices [7]. For example, GDPR defines what kind of data is considered to be personal data, and the work [5] summarizes privacy-sensitive data items in IoT contexts, and detects improper disclosures. In the meanwhile, some works focus on designing privacy-preserving mechanisms to enforce the regulations in IoT. For example, works [8], [9] propose solutions for privacy disclosure such as security and privacy labels for IoT devices. Other works [10] introduce methods to guide developers to design privacy compliant IoT applications. To achieve the privacy goal in such a scenario, it relies on the device vendors and app developers to provide such privacy disclosure mechanisms. However, there are challenges in existing solutions. First, there is a lack of unified privacy disclosure mechanisms for both IoT devices and companion apps, making it difficult for developers to correctly enforce the regulations in diverse IoT scenarios. Second, there are neither effective ways to dynamically notify end-users about current privacy collections, nor to dynamically configure the privacy settings of their devices.

Matter is an emerging IoT application layer standard [11] introduced by the Connectivity Standards Alliance (CSA), which provides unified mechanisms for better enhancing interoperability across devices and vendors. It has been widely adopted by large companies such as Google home [12], Apple Home [13], and Amazon Alexa [14]. Although Matter supports application layer secure connection, it by design does not directly support privacy disclosure mechanisms for product developers to design privacy-preserving applications.

To address the current challenges, in this position paper, we present the MATTERCOMPLIANCE framework based on Matter, leveraging its inherent interoperability across different devices and vendors. The key idea is to extend the current Matter Data Model to carry additional privacy disclosure, and provide interfaces (commands) for controllers and users to access and configure. The framework follows three design principles: 1) proactive privacy disclosure, 2) unify interfaces for developers that are easy to integrate with various devices, and 3) configurable privacy settings.

## II. BACKGROUND

### A. Matter Device Data Model

The architecture of Matter protocol is divided into layers for different responsibilities [11] (see Figure 1). To better support the functionality of the application layer-which contains logic to display the characteristics of IoT devices or control them-Matter introduces the Data Model layer which defines data and elements into structures that the application layer can operate on. To extend the Matter protocol to support proactive privacy

Workshop on Security and Privacy in Standardized IoT (SDIoTSec) 2025
24 February 2025, San Diego, CA, USA
ISBN 979-8-9919276-6-6
https://dx.doi.org/10.14722/sdiotsec.2025.23048
www.ndss-symposium.org

disclosure, it is necessary to understand how the device data is structured and managed.

In the Data Model layer, the specification defines elements and access qualities. Primary elements include fabric, node, endpoint and cluster. A fabric is a security domain in which set of nodes can be identified and communicate with each other by accessing Data Model elements. A node represents a unique instance of a Matter device in a fabric with an IP address. A full product such as a smart camera or speaker can be abstracted as a node. A node is a collection of one or more endpoints, where endpoints are used to indicate device types such as a sensor device or light device, and the different endpoints support different functions. Clusters are the lowest independent functional building block elements. Different endpoints contain collections of different clusters. A cluster defines a client and server side that correspond with each other during interactions. It contains elements including attributes, events and commands. The server cluster provides attribute data, events and commands, and the corresponding client cluster initiates interactions such as invoking commands.

Cluster attributes, events and commands shall define their access and privilege. The specification defines three types of access, including read, write and invoke access. Read access for an element means a client cluster can make a request for data values associated with it. Write access means a client cluster can request to modify attribute data. Invoke access enables a client cluster to execute a predefined command.

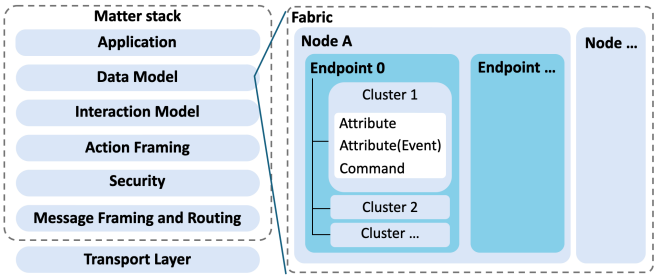


Fig. 1: Matter Protocol Stack and Data Model.

### B. Privacy Definition Under IoT

To achieve the goal of designing the privacy-preserving framework, it is essential to identify and define privacy-sensitive data within the IoT usage context. Following the approach of the previous study [5], we adopt the categories of privacy-sensitive data items in IoT devices, including device tracking data (*e.g.*, device id, hardware id, IP address), sensor data (*e.g.*, lock location, body weight, air quality) and device-attached data (*e.g.*, device name, camera image, turn on time). According to the GDPR [1], device tracking data and biometric data (a subset of sensor data) are classified as personal data because they can identify or represent a natural person. While other data items in these categories such as the device-attached data (*e.g.*, device metadata like device name and Bluetooth information, device usage data like door status and favorite shows, and timing data like turn on time, driving speed, and bed time) are not explicitly addressed in the regulation, prior works [15], [16], [17], [18] indicate that such information can be exploited by vendors or attackers to understand or infer user preferences and privacy norms or behaviors associated with these devices.

The clusters defined by the Matter protocol can include these three categories of privacy-sensitive data. For example, the Basic Information cluster includes attributes such as VendorID and ProductID, while the WiFi Network Diagnostics cluster contains BSSID, all of which are considered device tracking data. The Temperature Measurement cluster includes attributes like MeasuredValue, the Door Lock cluster includes attributes such as LockState, AutoRelockTime, and DoorState attributes, and the Window Covering cluster includes CurrentPositionLift. These attributes are considered sensor data or device usage data. Therefore, the Matter Data Model serves as a suitable foundation for privacy disclosure.

## III. Framework Design

### A. Overview

To achieve the privacy goal, we follow three key design principles when integrating privacy mechanisms into Matter-enabled systems: 1) enhancing privacy data disclosure mechanisms, 2) enabling aggregation across multiple devices, and 3) incorporating configurable privacy features. These principles ensure precision, transparency, scalability, and maintain compliance with current privacy regulations such as the GDPR [1].

First, the system is designed to provide accurate and proactive privacy disclosure by leveraging the Data Model of the Matter protocol. The Data Model layer is below the application layer, and organizes device data in a will-defined structure, enabling precise data collection. Unlike current solutions that are integrated by developers after app development, we propose a privacy compliance by design framework. that more focus on collecting privacy data usage on the application level and suffer from precision issues (*e.g.*, [5]), the Matter framework systematically transfers device data to controllers. Second, the system supports aggregation across multiple devices, which aligns with the multi-device nature of real-world IoT environments. While maintaining compatibility with the current Matter framework, the privacy design minimizes efforts to developers and provides flexibility to integrate or disable privacy-related features as needed. It ensures that developers can incorporate the privacy extensions into their products without significant modifications to their workflows, and even when for multiple devices usage scenarios. The ability to aggregate privacy data disclosure mechanism across devices enhances usability. Finally, the design incorporates configurable privacy features to help users better control their data. In compliance with current regulations, our design provides mechanisms to configure privacy settings (*e.g.*, disabling the collection of device location data). Matter provides such capabilities by commands. The controller can use commands to configure current privacy settings.

Following the three design principles, we propose the MatterCompliance framework (see Figure 2). On the device side,

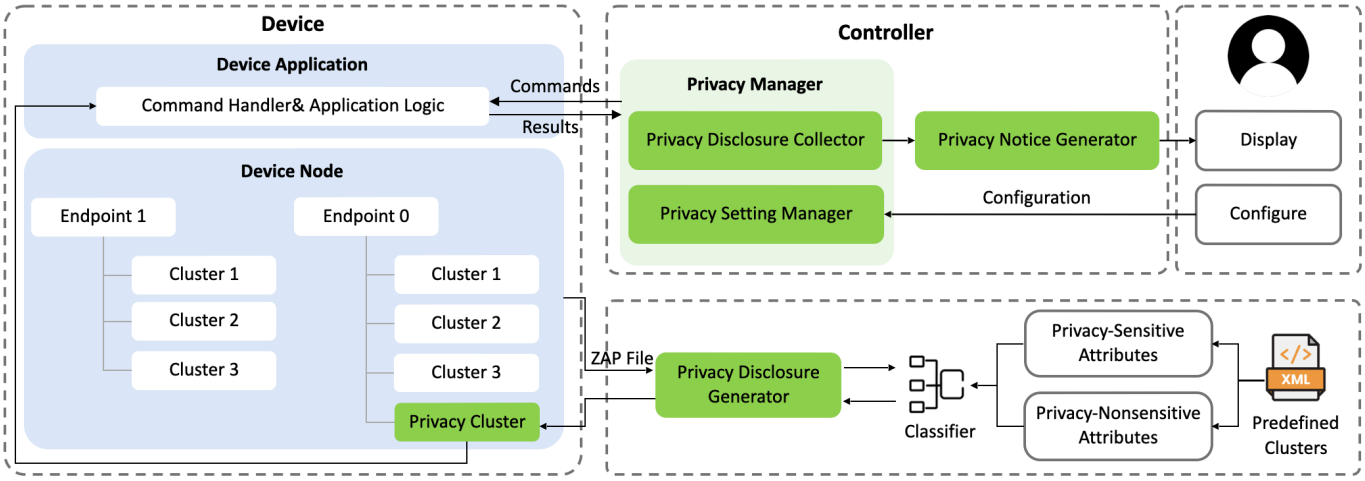


Fig. 2: MATTERCOMPLIANCE Framework Overview.

we introduce a new privacy cluster which contains all cluster attributes and indicates whether they are privacy-sensitive. To support configurable privacy settings, new commands related to the privacy clusters should be added, which introduce new application logic and handlers in the device application in the Matter stack. On the controller side, based on the user query, the component Privacy Notice Generator should be able to generate privacy notice and send to users. The Privacy Data Collector can use commands to read the attributes and values in the privacy cluster, and gather the privacy data for the Privacy Notice Generator. Users can also configure current privacy settings (*e.g.*, stop collecting the device location information), and the configurations will be replaced by commands to enforce the settings.

### B. Privacy Disclosure Generator

The Matter protocol supports incorporating custom clusters and commands, referred to as extensions. To implement these extensions, first we need to define the extended Data Model, including the basic information about the cluster, custom Matter attributes and commands. The definition should be placed in an XML file. For the custom privacy cluster, each attribute is defined by combining the cluster name with the attributes name (*e.g.*, "BasicInformation_VendorID"). The attribute value type is set to enum8 (see List 1 as attribute type design). All attributes are categorized into privacy-nonsensitive data and privacy-sensitive data. After defining the Data Model, privacy cluster related source code should be implemented. Methods including Read(), Write(), InvokeCommand() should be overridden to handle incoming commands. Additionally, necessary references to the cluster such as XML parsing tests, JSON file used by ZAP GUI, must be added, followed by regenerating all code. After all these steps, the privacy cluster can be added to the device's ZAP file by enabling it within ZAP, a tool used for configuring enabled clusters and commands.

```
enum PrivacyDisclosureEnum: enum8 {
    not-privacy-sensitive-data = 0;
    privacy-sensitive-data = 1;
}
```

Listing 1: Attribute Type Design.

In our privacy cluster design, we want to set the attributes and values to data items and the corresponding privacy disclosures. To do so, we introduce another component named Privacy Disclosure Generator. It first collects the current clusters and attributes. The clusters enabled by the device can be found in a ZAP file, which is a JSON format file containing enabled clusters selected by device vendors/developers. In this file, it might contain some of the predefined clusters, and vendor customized clusters. The Privacy Disclosure Generator collects all enabled clusters and the attributes. Then, the collected data will be sent to a classifier, which has already been trained to mark the input data as privacy sensitive data or nonsensitive data in IoT contexts. Predefined clusters and attributes in XML format in Matter specification which can be found in the Matter repository [19] can serve as materials to train the classifier.

### C. Privacy Cluster Design

The MATTERCOMPLIANCE framework introduces a novel privacy disclosure mechanism. This is achieved by the design principles outlined in subsection III-A. Following the second principle, the integrating component is designed to avoid significant implementation changes to the existing Matter framework, ensuring seamless integration with various vendors and devices. Additionally, the framework design should provide configurable capability to controllers and users.

To enable privacy disclosure, MATTERCOMPLIANCE introduces an additional cluster, referred to as the Privacy Cluster, which is under Endpoint 0. Endpoint 0 is reserved for the Utility Clusters that contain management and diagnostic features of a Matter node, and every Matter node includes such an endpoint. Utility Clusters serve special purposes, providing functionalities such as device paring and software update. Other clusters are Application clusters, which are specific to

different applications (*e.g.*, Door Lock cluster, Fan Control cluster). The Privacy Cluster can serve as a utility cluster to provide privacy disclosure. Matter protocol supports adding customized clusters building on top of the Matter stack, and the Privacy Cluster can be implemented this way. The process involves designing the cluster's attributes and values. This design requires first aggregating the attributes and values into an XML file to define the cluster data structures. Then the ZAP file of the device, which is a JSON file containing all enabled clusters, needs to be updated to include the new clusters. Finally, the modified ZAP file is used to generate the corresponding C++ code, which aggregate the newly defined Privacy Clusters into the system.

The Privacy Cluster by design is on the device side. To provide configurable capabilities to users, MATTERCOMPLIANCE introduces commands to help controllers and users to configure the current privacy practice. To support privacy-related commands, application layer handling logic should be implemented. Then, controllers will be able to access and control the privacy related data and settings through the predefined commands. After enabling the new clusters, the privacy disclosure collector (part of the privacy manager) should be able to read the attributes of the new Privacy Cluster, gather all privacy-sensitive data and send it to Privacy Notice Generator to generate user notice. Additionally, users can control the privacy setting manager to send commands to the privacy cluster, to adjust privacy settings in the real time (detailed design see subsection III-E).

### *D. Privacy Notice Generator*

To deliver privacy notice to users, we designed Privacy Notice Generator. In this paper, we use Alexa Skill as an instance to illustrate the Privacy Notice Generator design. The skill first asks the Component Privacy Data Collector to send commands to the device and retrieve the attributes and values of the privacy cluster, which contains all privacy-sensitive data related to the device. Next, the skill integrates Large Language Model (*i.e.* ChatGPT) to generate privacy notices using trained prompt and predefined template based on the the attributes and values of the privacy cluster. Moreover, we add serval *Intents* to capture users' verbal requests in the skill's front-end code and corresponding *IntentHandlers* to provide a response to user and execute skill functions in the backend code. This mechanism ensures a seamless and interactive way for users to access privacy-related information.

### *E. Privacy Settings Manager*

This subsection discusses how MATTERCOMPLIANCE satisfies the third design principle: enabling configurable privacy settings. To achieve this, several components are introduced, including the privacy setting manager, commands for the privacy cluster and associated command handlers and logic.

The privacy setting manager is located within the controller and takes user queries as inputs (*e.g.*, 'Stop collecting my device's product id'). Users may want to configure the current privacy settings (*e.g.*, whether to provide location information or VendorID to the controller), and the privacy setting manager processes these configure queries by sending corresponding commands to the device application layer, which enables the fine-grained management of specific privacy-sensitive data.

As illustrates in subsection III-C, commands to the Privacy Cluster and the corresponding application layer handler and logic should be implemented on the device side to provide configurable capabilities. Designing commands to control privacy settings requires considering three key conditions: 1) users must be able to control the collection of specific privacy-sensitive data at runtime; 2) managing privacy-sensitive data must not interfere with the device commissioning or secure connection process; 3) managing privacy-sensitive data must not disrupt the functional logic on the controller side. The reasons to the three requirements are as follows: First, enabling users to control the collection of privacy-sensitive data at runtime is convenient. If users are required to repair or power off the device to configure privacy settings, they may unwilling to do so. Therefore, updating the corresponding ZAP file and recompiling it into C++ code to configure privacy settings would create usability issues for end-users. Second, some privacy-sensitive data, such as vendor id, is closely tied to the commissioning process. Modifying this data could prevent the device from connecting to the Matter fabric. Third, the controller's functional logic, such as controlling a light or an air purifier, should be carefully managed. As incorrectly disabling such functions might lead to errors or crashes in the applications like the Google Home app.

We propose a design for commands and logic that adheres to these conditions. For the first condition, controlling the collection of data at runtime, the device application layer should implement command handlers (*e.g.*, the HandleDisableProductNameCommand function), logic and necessary callback functions to support commands. These commands will manage the values of corresponding attributes (*e.g.*, ProductName), disable entire privacy-sensitive cluster if needed, and prevent attribute values from being updated by the original logic. To meet the second and the third conditions, the framework introduces a blacklist mechanism to mark commissioning-related privacy-sensitive data and controller application-related functional data as uncontrollable (*e.g.*, users should not disable thermometer temperature or light on/off by commands). Items included in the blacklist should be validated through real-world device testing. The mechanism ensures users cannot take control of such data, maintaining the integrity of the device's core functions.

## IV. CONCLUSION

Our research introduces MATTERCOMPLIANCE, a privacy compliance framework within the Matter Protocol. It introduces a new Privacy Cluster and related components to provide a unified solution aligned with regulations and policies. The framework follows three principles: providing proactive privacy disclosure for IoT devices, unifying interfaces for developers to integrate across diverse IoT devices from different vendors, and enabling configurable privacy settings.

Additionally, we design privacy notices to users over voice and user interfaces to manage current privacy settings. For future work, we aim to develop additional mechanisms to enhance privacy-preserving features. These include richer privacy disclosure items, improved privacy notice for users, and more user-friendly configurations for greater control over privacy settings.

## REFERENCES


[1] "General data protection regulation," https://gdpr-info.eu/, 2024.

[2] "California consumer privacy act," https://oag.ca.gov/privacy/ccpa, 2024.

[3] J. Ren, D. J. Dubois, D. Choffnes, A. M. Mandalari, R. Kolcun, and H. Haddadi, "Information exposure from consumer iot devices: A multidimensional, network-informed measurement approach," in *Proceedings of the Internet Measurement Conference*, 2019, pp. 267–279.

[4] D. Y. Huang, N. Apthorpe, F. Li, G. Acar, and N. Feamster, "Iot inspector: Crowdsourcing labeled network traffic from smart home devices at scale," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 2, pp. 1–21, 2020.

[5] Y. Nan, X. Wang, L. Xing, X. Liao, R. Wu, J. Wu, Y. Zhang, and X. Wang, "Are you spying on me?{Large-Scale} analysis on {IoT} data exposure through companion apps," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 6665–6682.

[6] D. Schmidt, C. Tagliaro, K. Borgolte, and M. Lindorfer, "Iotflow: Inferring iot device behavior at scale through static mobile companion app analysis," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 681–695.

[7] P. Liu, S. Ji, L. Fu, K. Lu, X. Zhang, J. Qin, W. Wang, and W. Chen, "How iot re-using threatens your sensitive data: Exploring the user-data disposal in used iot devices," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2023, pp. 3365–3381.

[8] P. Emami-Naeini, Y. Agarwal, L. F. Cranor, and H. Hibshi, "Ask the experts: What should be on an iot privacy and security label?" in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 447–464.

[9] P. Emami-Naeini, J. Dheenadhayalan, Y. Agarwal, and L. F. Cranor, "An informative security and privacy "nutrition" label for internet of things devices," *IEEE Security & Privacy*, vol. 20, no. 2, pp. 31–39, 2021.

[10] N. Alhirabi, S. Beaumont, J. T. Llanos, D. Meedeniya, O. Rana, and C. Perera, "Parrot: Interactive privacy-aware internet of things application design tool," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 7, no. 1, pp. 1–37, 2023.

[11] "Matter," https://csa-iot.org/all-solutions/matter/, 2024.

[12] "Google home app," https://home.google.com/welcome/, 2024.

[13] "Apple Home App," https://www.apple.com/home-app/, 2024.

[14] "Amazon Alexa," https://developer.amazon.com/en-US/alexa, 2023.

[15] G. Chu, N. Apthorpe, and N. Feamster, "Security and privacy analyses of internet of things children's toys," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 978–985, 2018.

[16] T. Davenport and J. Lucker, "Running on data: Activity trackers and the internet of things," *Deloitte Review*, vol. 16, pp. 5–15, 2015.

[17] N. Apthorpe, Y. Shvartzshnaider, A. Mathur, D. Reisman, and N. Feamster, "Discovering smart home internet of things privacy norms using contextual integrity," *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies*, vol. 2, no. 2, pp. 1–23, 2018.

[18] N. Abdi, X. Zhan, K. M. Ramokapane, and J. Such, "Privacy norms for smart home personal assistants," in *Proceedings of the 2021 CHI conference on human factors in computing systems*, 2021, pp. 1–14.

[19] "Matter Repository," https://github.com/project-chip/connectedhomeip/tree/master/data_model/1.4/clusters, 2024.