PICO: Privacy-Preserving Access Control in IoT Scenarios through Incomplete Information

Savio Sciancalepore
Eindhoven University of Technology
s.sciancalepore@tue.nl

Nicola Zannone
Eindhoven University of Technology
n.zannone@tue.nl

ABSTRACT

Internet of Things (IoT) platforms typically require IoT devices and users to provide fine-grained information to determine whether access to resources and services can be granted. However, this information can be sensitive for users and its disclosure can lead to severe privacy threats, forcing users to decide between receiving a service or protecting their privacy. To close this gap, this work proposes PICO, a framework for privacy-preserving access control in IoT scenarios through incomplete information. PICO allows IoT devices to evaluate the privacy risks of disclosing the information needed to access a service and determine at which level of granularity such information can be disclosed. At the same time, PICO empowers IoT platforms to evaluate access control policies even when incomplete information is provided and possibly grant access to services based on a customized service-dependent risk factor. Through simulations using data from real IoT devices, we show the existence of a trade-off between privacy and energy consumption on IoT devices running PICO, and that more privacy can be achieved for such devices only by sacrificing a consistent portion of the overall energy capacity.

CCS CONCEPTS

• **Security and privacy** → **Access control**; **Privacy-preserving protocols**; • **Computer systems organization** → **Sensor networks**;

KEYWORDS

Privacy-Preserving Access Control; Client Privacy; IoT Security; Disclosure Risk; Energy-Privacy Trade-off.

ACM Reference Format:
Savio Sciancalepore and Nicola Zannone. 2022. PICO: Privacy-Preserving Access Control in IoT Scenarios, through Incomplete Information. In *The 37th ACM/SIGAPP Symposium on Applied Computing (SAC '22), April 25–29, 2022, Virtual Event, .* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3477314.3508379

1 INTRODUCTION

Internet of Things (IoT) devices are nowadays increasingly employed in a variety of application domains, such as Smart Buildings [16], Smart Cities, and Smart Transportation Systems [29], pervasively accessing the users' environments and most intimate spheres.

The data acquired from IoT devices are increasingly used to allow users to access services and resources in a personalized and optimized fashion, e.g., based on their location and contextual information. This is the case, e.g., of smart illumination systems in Smart Buildings, intelligent traffic management systems in Smart Cities, and smart toll stations in Smart Highways. On the one hand, the IoT platform needs fine-grained information about the users and their physical context, so as to enhance users' experience. On the other hand, information such as address, real-time location or other personally identifiable attributes can be sensitive, and their disclosure can lead to severe large-scale privacy threats [17, 25].

Current access control systems adopted in IoT-based ecosystems force users to disclose fine-grained information, without offering any privacy-friendly alternative. A few works have proposed privacy-preserving access control solutions for IoT devices and their users [14, 30], but they typically rely on techniques that are too energy-demanding for constrained IoT devices [15], or outsource part of the computation to third parties [4, 11, 28], which might be compromised or not fully trusted [23, 29].

The challenge is to devise a privacy-preserving access control framework for IoT able to balance the security needs of IoT platforms and users' privacy, while ensuring its applicability to constrained IoT devices. In particular, users should be able to determine the risk of disclosing their information and decide at which level of granularity it can be disclosed to access a service, while the IoT platform should be able to assess the security implications of this choice.

Contribution. In this work, we propose PICO, a novel framework enabling privacy-preserving access control in IoT scenarios through the use of incomplete information. PICO allows IoT devices to evaluate the privacy risks associated with the information to be disclosed for policy evaluation, and to determine at which level of granularity such information can be disclosed when requesting access to a service. To mitigate the risks of an IoT platform to inadvertently grant (deny) access to unauthorized (authorized) users due to the use of incomplete information, we define the components and functions necessary to evaluate access control policies using coarse-grained information, where access is granted only if the risks are tolerable. We evaluated the performance of PICO via simulations using data available from real IoT devices, quantifying the trade-offs between privacy and energy consumption characterizing the adoption of different operational strategies supported by our solution. We showed that PICO allows IoT devices to access services using very little energy (0.3% of overall battery capacity), while ensuring an acceptable privacy level. IoT devices can increase their privacy at the cost of increased energy consumption, where the achievable privacy depends on the energy constraints of the IoT device.
Overall, PICO offers to IoT devices the unique opportunity to trade-off between privacy and energy consumption, enhancing situational awareness and enabling the selection of the attribute disclosure strategy that best fits energy availability and delay requirements.

SAC '22, April 25–29, 2022, Virtual Event,

The paper is organized as follows. Sec. 2 introduces our motivations and requirements, Sec. 3 provides the details of PICO, Sec. 4 validates PICO through experiments, Sec. 5 discusses related work and cross-compares PICO with them; finally, Sec. 6 concludes the paper.

2 MOTIVATIONS AND REQUIREMENTS

This section introduces the target IoT scenario along with the underlying assumptions (Sec. 2.1) and reference use cases motivating the design of PICO (Sec. 2.2). Based on the challenges emerging from the use cases, we identify the main requirements for a privacy-preserving access control solution for IoT (Sec. 2.3).

2.1 Reference Scenario and Assumptions

In this work, we address an IoT scenario where access to services and resources is managed via the Attribute-Based Access Control (ABAC) paradigm. Thus, to access a service, users have to demonstrate the possession of valid attributes, satisfying the policy associated to the requested service. In line with traditional access control architectures, we assume that the entities in the system (the users and/or their IoT devices) demonstrate the possession of specific attributes by providing dedicated proofs, such as tokens, in the form of digital objects signed by a Trusted Third Party (TTP).

Moreover, we assume that the attributes to be provided for allowing service provisioning can be sensitive for the requester and/or the attribute providers. Such sensitive information include, e.g., user's location, trajectory, personal preferences, and health data. In addition, we consider use cases where specific information might not be available to the user because of access restrictions, e.g., information regarding third-party-owned transported items.

We also assume that IoT devices are energy constrained. Specifically, in line with common assumptions and capabilities, we assume that the IoT devices can support the execution of traditional symmetric (e.g., AES) and asymmetric cryptography (e.g., RSA and ECC) solutions, but they cannot afford more demanding techniques, such as the ones based on pairing (e.g., Attribute-Based Encryption (ABE)), due to their high energy and bandwidth requirements [15].

In this light, we pursue a different approach in which users can disclose information at a higher level of granularity to protect their privacy. However, making an access decision using course-grained or incomplete information might result in the access control mechanism granting access to unauthorized users. In such cases, traditional access control mechanisms simply deny access to services/resources when the information needed for policy evaluation is missing, often forcing users to decide between receiving the service or protecting their privacy. The challenge is, thus, to devise a privacy-preserving access control framework able to balance the service provider's security requirements and users' privacy.

2.2 Use cases

To illustrate the challenges to be addressed for the design of our framework, we provide a few use cases involving access control in IoT scenarios, requiring users to share *sensitive* information.

Use Case 1 (Smart Cities 1). *Let us assume a* Smart City*, where several Roadside Units (RSUs) are deployed throughout the roads of the city and interact with users and vehicles to enable several services, ranging from the management of traffic lights to road illumination. In particular, we assume that road illumination is managed based on the presence/proximity of users (either pedestrians or vehicles). Specifically, the Smart City enforces a policy for each (group of) light(s) deployed on a road, such that lights are turned on only if at least one user is located within a given radius from the lights. For enhanced user safety, RSUs not only turns on the lights closest to the users' location but also other lights in the vicinity, trying to anticipate their movement to increase the visibility on the road. When approaching a given street, the user requests the closest RSU to turn on the lights on the street. In turn, the RSU requires the user to provide its location, final destination and other relevant information and, based on this information, it decides whether a group of lights should be turned on. However, user's location and destination are sensitive pieces of information as their knowledge could enable the RSU to track the user' movement, as well as to infer sensitive information (visited locations, personal preferences, etc.). Therefore, users could decide not to disclose precise information, but to obfuscate it, e.g., by providing a region. While allowing a user to disclose* obfuscated *information protects the location privacy of the user, the RSU should take the decision on turning on (or not) the lights based on the information in its possession. This introduces uncertainty in policy evaluation, resulting in the RSU potentially turning on the wrong group of the lights. In particular, the RSU can only determine whether at least one user is expected to be in the range of the lights to be turned on only with a certain probability.*

Use Case 2 (Smart Cities 2). *Let us assume the* Smart City *of Use Case 1, where dedicated RSUs interact with users and vehicles to manage restricted traffic area. We assume that the Smart City encompasses limited traffic zones to protect the city centre from excessive traffic and reduce pollution. In particular, access to limited traffic zones by (motor-powered) vehicles is limited to certain hours of the day and otherwise are protected by smart barriers. Outside the allowed hours, access to each zone is only permitted to residents and for the delivery of goods to stores within the zone. When a vehicle approaches the barriers at the border of a limited traffic zone, the RSU controlling the barriers requests the vehicle to provide the information needed for policy evaluation (either the residence address or the shipping address along with a valid delivery note) and, based on the provided information, the RSU decides whether to lower the barriers. However, this information can be regarded as sensitive and could enable the Smart City to track users' movements, as well as to infer their behavioral patterns. Therefore, users could decide not to disclose the precise address but to only provide the area code. Thus, the RSU should take a decision on whether lowering the barrier (or not) based on the information in its possession. This could be problematic, for instance, when the area code provided by the user overlaps with the restricted limited traffic zone, leaving the RSU with the risk of inadvertently granting access to unauthorized users.*

Use Case 3 (Smart Highway). *We assume a* Smart Highway Payment System*, where users driving vehicles in the highway interact with smart RSUs located at the entry/exit points of the highway to provide the information needed for toll payment. The road fee is calculated based on the distance between the entry and exit points as well as on the characteristics of the vehicle such as its dimension and weight. By default, the RSU applies the maximum fee, i.e., the one corresponding to the maximum weight category and the maximum possible distance. Based on the available information, such a fee could be decreased using ad-hoc discount rules, up to potentially being equal*

to the fee that the vehicle would have paid if all required information is available. When a vehicle approaches, the RSU asks for the full details of the vehicle, including its brand and model (to determine its dimension), weight, and the entry point (to compute the travel distance). The disclosure of this information, however, can rise privacy concerns for the user. To limit the exposure of potentially identifiable information (e.g., vehicle brand and model, entry point), a user might decide to disclose only an approximation of the vehicle's weight and travel distance, for instance in terms of an interval range, without disclosing the exact weight and entry point. Using the provided information, the RSU should decide whether to apply (or not) a specific discount on the initial toll road fee. This could be challenging as, for instance, the weight interval provided by the user might not fall entirely in any of the vehicle weight classed defined by the Highway Payment System. Thus, the system could overcharge users or treat users who did the exact same road trip differently, leading to complaints and legal actions.

Use Case 4 (Smart Building). *Let us assume a Smart Building, comprising several apartments located in different corridors on different floors. Common areas in the building are regulated and managed by an IoT platform through which tenants can, e.g., turn on/off lights in the corridors. However, we assume that tenants are only allowed to turn on the lights in the corridor where their apartment is located. A tenant might disclose her apartment number when requesting to turn on/off the lights but this can pose privacy issues to the tenant, as this information could allow her direct identification as well as enable the inference of sensitive situations (e.g., whether the tenant is in the apartment or she is away). To protect her privacy, the tenant might disclose information about her apartment at higher level of granularity. For instance, she might only disclose the corridor or the floor in which her apartment is located. However, while providing coarse-grained information can preserve the tenant's privacy, it has an impact on policy evaluation. In particular, depending on the level of granularity the information is provided (and the access constraints specified in the policy), the IoT platform can make a decision with a different level of confidence. For instance, if only the apartment floor is disclosed, there is the risk that an incorrect access decision is taken (e.g., based on the number of apartments on the floor).*

Use Case 5 (Smart Transportation Systems). *We assume a Smart Transportation System, e.g., a ship, carrying a number of containers, each including different packages and goods handled by a given shipping agency. We assume that the containers feature an IoT device, allowing the container to communicate and exchange data with the Smart Transportation System platform. We also assume that the IoT device of the container holds information about the packages stored within the container, their source and destination, and other information about the content of the packages, in line with the information provided by the respective sender. Assume there is an emergency onboard, where the temperature of the area where the container is located exceeds its regular value. In this case, the platform can ask information about the type of goods stored in the container, to decide whether to move the container to a different location. However, to protect their privacy, users might have provided to the shipping agency only limited and coarse-grained information about the packages, or might not have authorized the shipping agency to share such information with third-parties. Although providing coarse-grained information can preserve the sender's privacy, it can result in a financial loss for the sender and the ship. Depending on the level of granularity the information is provided, the Smart Transportation System can authorize or not the relocation of the container. For instance, not disclosing information about the presence of frozen or perishable goods in the container might lead the Transportation System platform to conclude that moving the container is not necessary, causing the wasting of the goods.*

2.3 Discussion and Requirements

The use cases show that the IoT platform's demand of sensitive information for policy evaluation can raise privacy concerns. In particular, the IoT platform and individuals can have conflicting requirements: the IoT platform requires accurate information for making a reliable access decision, while individuals seek their own privacy by avoiding the disclosure of sensitive fine-grained attribute values.

The design of an access control solution that preserves *client privacy* should resolve those conflicting requirements. Specifically, the access control mechanism should not force the IoT device requesting for a service to disclose sensitive information, which can lead to user identification and profiling (e.g., the residence address in Use Case 2 or the apartment number in Use Case 4) but should give the IoT device (on behalf of the user) the choice of which information to disclose (*No disclosure sensitive info.*). For instance, in Use Case 3 the weight and distance ranges can be decided by users on a case-by-case basis to guarantee a different level of privacy. To this end, the solution should provide the IoT device with mechanisms to evaluate the risks associated with the disclosure of a given attribute value and determine at which level of granularity attribute values can be disclosed to guarantee a certain privacy level (*Disclosure Risk (Client)*). At the same time, as illustrated by the use cases in Sec. 2.2, determining whether access to resources should be granted or not when the primary information necessary for policy evaluation is unavailable or known only to a limited extent, can expose the IoT platform to the risks of taking the wrong decision. Therefore, the access control mechanism should be equipped with features for quantifying the impact of the use of coarse-grained information in the policy evaluation process (*Server Risk Assessment*). In addition, due to the frequent breaches of information and metadata occurring on Internet servers [17, 25], IoT devices should be able to work locally, avoiding the disclosure of sensitive information even to potentially trusted servers (*No computations outsourcing*).

As shown in the literature [7, 13], guaranteeing privacy within IoT scenarios can come at the cost of an increasing *energy consumption*. This is problematic due to the limited energy available on IoT devices. Therefore, a solution for privacy-preserving access control should be evaluated also in terms of energy consumption (*Evaluation*), and should allow users to select strategies accounting for a trade-off between energy consumption and privacy guarantees, based on the amount of residual energy (*Privacy Trade-off*).

Complementary to the problem of guaranteeing users' privacy, it is desirable for an access control mechanism to also ensure the so-called *server privacy* [28, 30], which requires that also policies are protected (*Policy Confidentiality*). Indeed, the disclosure of the security policies employed might reveal confidential information about a company's business strategies and its commercial relations as in Use Case 5 or leak sensitive information about the resource owner [22].

In the next section, we present the foundations for the design and development of a privacy-preserving access control framework that meets these requirements; then, in Sec. 5 we use the identified requirements to evaluate the capability of competing solutions.

3 PRIVACY-PRESERVING ACCESS CONTROL THROUGH INCOMPLETE INFORMATION

This section presents PICO, our novel privacy-preserving access control framework supporting a client to access a service even when it provides coarse-grained information to protect its privacy.

3.1 Architecture Overview

Fig. 1 provides the logical architecture of PICO. The architecture comprises a *client* and a *server*: the client is the entity requesting a service and the server is the entity controlling the service, who grants access only if the client meets certain access constraints. In line with ABAC, access constraints (referred to as *policy*) are specified in terms of attributes. The *Attribute Provider (AP)* is a TTP that collects and certifies the client's attribute values, in the form of signed objects, e.g., tokens. Thus, the server also interacts with the AP to (obtain the cryptography materials to) verify their validity.

When providing attribute values and evaluating the policy using such values, the client and the server rely on a shared representation of the attributes values, encoded in a *data model*. Specifically, the data model encodes the attributes' domain in a tree-based structure, from the most general (root) to the most specialized values (leaves), along with the semantic closeness between attribute values. The data model is also used by the AP to guarantee that attribute values are valid even when disclosed at a higher level of granularity. More details on the data model are provided in Sec. 3.2.

Within the client and the server, we identify a number of components. Note that these components can be either all integrated in a single IoT device or distributed among different physical machines.

The client comprises the *Attribute Handler (AH)*, which is responsible for the local storage and management of attributes (retrieved from the AP). Within the AH, the *Decision Risk Assessment (DRA)* module assesses the risk associated to the disclosure of attribute values and decides at which level of granularity information can be disclosed. The DRA module is described in Sec. 3.3.

Within the server, the *Access Control Manager (ACM)* is the component responsible for the protection of sensitive resources and services. This component comprises three modules that resembles the logical elements of the XACML reference architecture [12], i.e., the *Probabilistic Policy Decision Point (PPDP)*, *Policy Administration Point (PAP)* and *Risk Resolution (RR)* modules. The PPDP is responsible for policy evaluation. In particular, based on the set of attributes provided by the client and the access control policies fetched from the PAP, the PPDP computes likelihood estimations of decisions, representing the risks of incorrectly granting/denying access to the resource due to incomplete information (cf. Sec. 3.4). For the scope of this work, the PAP mainly acts as the policy repository, but it can also provide the resource owner with interfaces for policy specification and management. The RR module is the component responsible for policy enforcement, similarly to the Policy Enforcement Point (PEP) in the XACML architecture. Based on the likelihood estimations provided by the PPDP, this module determines whether

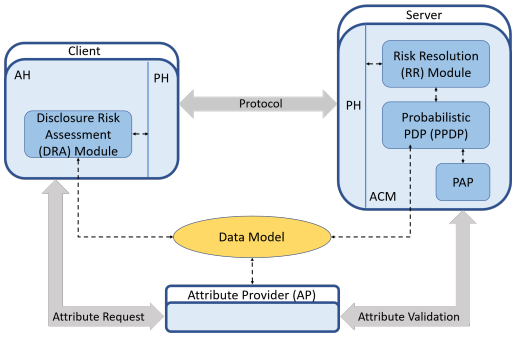

Figure 1: Overview of the architecture of PICO.

the associated risks are lower than a given tolerance threshold and, in case, grants the client access to the resource (cf. Sec. 3.5). Both client and server encompass a *Protocol Handler (PH)*, which supports the interactions with the other party (cf. Sec. 3.6).

In the remainder of the section, we provide the detailed logic and operations of the main components of PICO.

3.2 Data Model

PICO employs a shared *data model*, encoding the attribute values and their relations. The function of the data model is twofold. First, the data model supports the client in determining the level of granularity at which attribute values can be disclosed (cf. Sec. 3.3). Second, it supports the server in assessing the risk of taking incorrect decisions when the information needed for policy evaluation is incomplete (cf. Sec. 3.4). Next, we provide the formalization of the data model along with additional concepts used through the paper.

Let $\mathcal{A} = \{a_1, \ldots, a_n\}$ be a finite set of attributes and, given an attribute $a_i \in \mathcal{A}$, $\mathcal{V}_{a_i}$ denote the domain of a_i, i.e., the set of possible values that a_i can assume. For the purpose of this work, we assume that attribute values can be specified at different levels of granularity, which can be captured through an attribute hierarchy.

Definition 1 (Attribute Hierarchy). *Given an attribute $a \in \mathcal{A}$, the attribute hierarchy H_a is a tuple (N, S, λ) where:*

- *N is the set of nodes, each representing a value in $\mathcal{V}_a$;*
- *$S \subseteq N \times N$ represents the specialization relation on N;*
- *$\lambda : S \rightarrow [0, 1]$ is a labeling function that defines the semantic closeness between two attribute values.*

Fig. 2 presents an example attribute hierarchy for attribute `apartment` in Use Case 4, where the leaf nodes represent the individual apartments and the root node ($\top$) represents all apartments in the building. Hereafter, we use notation $(a, v_i) \subset (a, v_j)$ to denote that, given an attribute a, v_i is a specialization of v_j where $v_i, v_j \in \mathcal{V}_a$. The specialization relation is reflexive, antisymmetric and transitive.

Labels are assigned to specialization relations to denote to what extent the knowledge of an attribute value allows one to infer another attribute value. One can observe that, if a subject has an attribute a with value v, the generalization(s) of v can also be associated with that subject. For instance, in Use Case 4, the tenant can disclose the apartment number when requesting to turn on the lights in her corridor; in this case the IoT platform can infer in which corridor and floor the apartment is located. However, the opposite might not hold; if IoT platform knows only the floor in

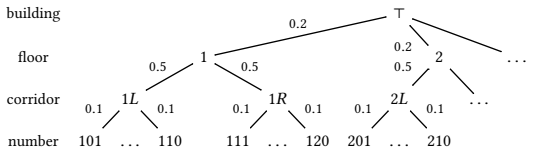

Figure 2: Hierarchy for attribute `apartment` in Use Case 4

which the apartment is located, it does not know the exact apartment in which the tenant lives. The IoT platform can only make an estimation of the apartment with a certain probability.

The form of the labeling function λ depends on the specific attribute. For instance, the semantic closeness between a value of attribute `apartment` and its specializations can be defined based on the number of child nodes, as illustrated in Fig. 2. Instead, for locations, which are generalized by taking larger regions, the semantic closeness can be defined as the portion of the generalized location covered by the specialized location.

The attribute hierarchy is used to compute the *degree of similarity* between two attribute values by extending the semantic closeness defined by the labeling function λ to account for the transitivity of the specialization relation. Given two attribute values (nodes) in the attribute hierarchy, their similarity is computed based on the paths connecting the two nodes.

Definition 2 (Degree of similarity). *Given an attribute $a \in \mathcal{A}$, let $H_a = (N, S, \lambda)$ be an attribute hierarchy for a. Given two nodes $n_i, n_j \in N$, the degree of similarity of n_i with n_j, denoted as $\sigma(n_i, n_j)$, is defined as per the following cases:*

Base case: *The degree of similarity of n_i with an adjacent node n_j and with n_i itself is:*

$$\sigma(n_i, n_j) = \begin{cases} 1 & \text{if } n_i = n_j \\ 1 & \text{if } (n_i, n_j) \in S \\ \lambda(n_i, n_j) & \text{if } (n_j, n_i) \in S \end{cases} \quad (1)$$

Path: *Let $\langle n_i, \ldots, n_j \rangle$ be a path from n_i to n_j. The degree of similarity of n_i with n_j is:*

$$\sigma(n_i, n_j) = \prod_{k=1}^{m-1} \sigma(n_k, n_{k+1}) \qquad (\text{with } n_i = n_1 \text{ and } n_j = n_m) \quad (2)$$

It is worth noting that similarity is not symmetric, i.e., $\sigma(v_i, v_j)$ can be different from $\sigma(v_j, v_i)$, due to the antisymmetry of the specialization relations (see above). Also, the degree of similarity of a node with itself is set to 1. This allows the PPDP to deal with the case where the attribute value specified in the policy is provided for policy evaluation, which should result in a "full" match.

3.3 Disclosure Risk Assessment

Entities can disclose their attributes at a different level of granularity, each of them revealing different amounts of information about the entity. To determine and quantify at which level of granularity the values of an attribute can be disclosed, we introduce the notion of *disclosure risk*. Intuitively, the disclosure risk for an attribute value represents the likelihood of inferring the exact attribute value of a user (i.e., the most specialized value) from that value. Based on such intuition, we compute the disclosure risk in terms of attribute similarity. We formalize such an intuition in the following definition.

Definition 3 (Disclosure Risk). *Let $H_a = (N, S, \lambda)$ be the hierarchy for an attribute $a \in \mathcal{A}$. Given a node $n \in N$ representing the exact attribute value (i.e., a leaf node in H_a) and its generalization n' (i.e., $(a, n') \subset (a, n)$), the disclosure risk of n', $\delta_{H_a}(n')$, is defined as:*

$$\delta_{H_a}(n') = \sigma(n', n) \quad (3)$$

From Eq. 3, it is easy to observe that the disclosure of the exact attribute value has a disclosure risk equal to 1, and that the disclosure risk decreases as the distance (in terms of path length in the hierarchy) between the disclosed value and the exact value increases.

We assume that a user is willing to disclose an attribute value v only if the disclosure risk associated with v is lower than an *attribute risk tolerance* τ_a given by the user for a, i.e., $\delta_{H_a}(v) < \tau_a$, with $\tau_a \in [0, 1]$. Based on such observations, we introduce the notions of *sensitive* and *non-sensitive* attribute value.

Definition 4 (Sensitive Attribute Value). *Let a be an attribute with attribute hierarchy $H_a = (N, S, \lambda)$. An attribute value $n \in N$ is referred to as* sensitive *if the disclosure risk of n is equal or greater than the attribute risk tolerance τ_a, i.e., $\delta_{H_a}(n) \geq \tau_a$. Otherwise, n is considered to be* non-sensitive.

We now describe the procedure to determine which values of an attribute can be disclosed (being *non-sensitive*) given the attribute hierarchy and the user's risk tolerance based on the disclosure risk. Let v be the exact value of the attribute a for a user (i.e., a leaf node in the attribute hierarchy H_a). In a bottom-up fashion, the procedure traverses the attribute hierarchy from v and computes the disclosure risk of the values generalizing v (recall that by definition the disclosure risk of v is equal to 1). The procedure terminates when a value whose risk disclosure is lower than the given risk tolerance is found or when the root of the attribute hierarchy is reached. From this procedure, it is easy to observe that all generalizations of a non-sensitive value are also non-sensitive.

It is worth noting that, when a set of attribute values is disclosed, we associate to the set the disclosure risk of the attribute value with the highest disclosure risk. Formally (abusing the notation), given a set of attribute values $V = \{v_1, \ldots, v_n\}$, one for each attribute $a_1, \ldots, a_n$, the risk disclosure of V is $\delta(V) = \max\{\delta_{H_{a_i}}(v_i) \mid v_i \in V\}$.

3.4 Probabilistic PDP

To compute the risk of inadvertently granting access to unauthorized users (or denying access to authorized user) when information needed for policy evaluation is incomplete, we employ an adaptation of the framework in [16]. This framework supports the probabilistic evaluation of ABAC policies and computes likelihood estimations of decisions based on the similarity between attribute values encoded in an attribute hierarchy.

In ABAC, policies and access requests (hereafter referred to as *queries*) are expressed in terms of attributes. Given a set of attributes $\mathcal{A}$, a query $q = \{(a_1, v_1), \ldots, (a_k, v_k)\}$ is a set of attribute name-value pairs (a_i, v_i) such that $a_i \in \mathcal{A}$ and $v_i \in \mathcal{V}_{a_i}$. The set of queries that can be built over $\mathcal{A}$ is $Q_{\mathcal{A}} = \wp(\bigcup_{a_i \in \mathcal{A}} \bigcup_{v_j \in \mathcal{V}_{a_i}} (a_i, v_j))$ (Given a set X, $\wp(X)$ denotes the powerset of X).

For the specification of ABAC policies, we use the following grammar, which resembles the XACML policy language syntax [12]:

$$\begin{aligned} p &= 1 \mid 0 \mid (t, p) \mid p_1 \triangledown p_2 \mid p_1 \triangle p_2 \\ t &= (a, v) \mid \neg t_1 \mid t_1 \wedge t_2 \mid t_1 \vee t_2 \end{aligned}$$

Table 1: Target and Policy Semantics

$[\![(a,v)]\!]_{\mathrm{T}}(q)$	=	$\max\{\sigma(v_i, v) \mid (a, v_i) \in q\}$
$[\![\neg t]\!]_{\mathrm{T}}(q)$	=	$1 - [\![t]\!]_{\mathrm{T}}(q)$
$[\![t_1 \vee t_2]\!]_{\mathrm{T}}(q)$	=	$[\![t_1]\!]_{\mathrm{T}}(q) + [\![t_2]\!]_{\mathrm{T}}(q) - [\![t_1]\!]_{\mathrm{T}}(q)[\![t_2]\!]_{\mathrm{T}}(q)$
$[\![t_1 \wedge t_2]\!]_{\mathrm{T}}(q)$	=	$[\![t_1]\!]_{\mathrm{T}}(q)[\![t_2]\!]_{\mathrm{T}}(q)$
$[\![1]\!]_{\mathrm{P}}(q)$	=	$(1, 0, 0)$
$[\![0]\!]_{\mathrm{P}}(q)$	=	$(0, 1, 0)$
$[\![(t,p)]\!]_{\mathrm{P}}(q)$	=	$[\![t]\!]_{\mathrm{T}}(q) \cdot [\![p]\!]_{\mathrm{P}}(q) + (1 - [\![t]\!]_{\mathrm{T}}(q)) \cdot (0, 0, 1)$
$[\![p_1 \triangledown p_2]\!]_{\mathrm{P}}(q)$	=	$(\ell_1^1 + \ell_2^1 - \ell_1^1\ell_2^1, \ell_1^0\ell_2^0 + \ell_1^0\ell_2^\perp + \ell_1^\perp\ell_2^0, \ell_1^\perp\ell_2^\perp)$
$[\![p_1 \triangle p_2]\!]_{\mathrm{P}}(q)$	=	$(\ell_1^1\ell_2^1 + \ell_1^1\ell_2^\perp + \ell_1^\perp\ell_2^1, \ell_1^0 + \ell_2^0 - \ell_1^0\ell_2^0, \ell_1^\perp\ell_2^\perp)$

where p denotes policies and t denotes the policy target. A target t can be *atomic*, denoted by (a, v) with $a \in \mathcal{A}$ and $v \in \mathcal{V}_a$, or *composite*. A composite target is a Boolean expression built over atomic targets using the set of operators $\{\neg, \wedge, \vee\}$. A policy p can be a single decision (hereafter called *atomic policy*), either *permit* (1) or *deny* (0), a *target policy* (t, p) or a *composite policy* in which two policies are combined using a policy combining operator. In this work, we consider two binary operators, $\triangledown$ and $\triangle$, which correspond to the XACML policy combining algorithms *permit-overrides* and *deny-overrides* respectively [9], but they can be trivially extended to deal with an arbitrary number of policies/targets (see [16]). These operators are defined on the set $\{1, 0, \perp\}$, where $\perp$ represents the *not-applicable* decision, as follows: $x \triangledown y = \perp$ when both x and y are equal to $\perp$, $x \triangledown y = 1$ when either x or y is equal to 1, and $x \triangledown y = 0$ otherwise. Conversely, $x \triangle y = \perp$ when both x and y are equal to $\perp$, $x \triangle y = 0$ when either x or y is equal to 0, and $x \triangle y = 1$ otherwise. Hereafter, we denote the set of targets and policies that can be constructed over $\mathcal{A}$ as $\mathcal{T}_{\mathcal{A}}$ and $\mathcal{P}_{\mathcal{A}}$ respectively.

The evaluation of a policy requires evaluating the target (if present) to determine the applicability of the policy to a query. Given the set of targets $\mathcal{T}_{\mathcal{A}}$ and the set of queries $Q_{\mathcal{A}}$, the evaluation function $[\![\cdot]\!]_{\mathrm{T}} : \mathcal{T}_{\mathcal{A}} \times Q_{\mathcal{A}} \rightarrow [0, 1]$ determines the applicability of a target t to a query q such that $[\![t]\!]_{\mathrm{T}}(q)$ is the likelihood that q matches t based on the degree of similarity between the attribute values in the query and the ones specified in the target. Targets evaluate to a value in the range $[1, 0]$ where 1 indicates that the target certainly matches the query and 0 that the target certainly does not match the query. Accordingly, it is trivial to observe that the likelihood that a target t does not match a query q is defined as $1 - [\![t]\!]_{\mathrm{T}}(q)$. The target semantics is formally defined at the top of Table 1.

Given the set of policies $\mathcal{P}_{\mathcal{A}}$ and the set of queries $Q_{\mathcal{A}}$, the evaluation function $[\![\cdot]\!]_{\mathrm{P}} : \mathcal{P}_{\mathcal{A}} \times Q_{\mathcal{A}} \rightarrow [0, 1]^3$ computes the likelihood that a certain decision is returned based on the degree of similarity between attributes such that, given a policy p and a query q, $[\![p]\!]_{\mathrm{P}}(q) = (\ell^1, \ell^0, \ell^\perp)$ where ℓ^1 denotes the likelihood that the decision is *permit*, ℓ^0 the likelihood that the decision is *deny* and $\ell^\perp$ the likelihood that the decision is *not-applicable*. The sum of these likelihoods is equal to 1, i.e., $\ell^1 + \ell^0 + \ell^\perp = 1$. The policy semantics is formally defined at the bottom of Table 1. Note that, even if the evaluation of both targets and policies uses $\{1, 0\}$, these values have different meanings, which should be always clear from the context.

3.5 Risk Resolution

The evaluation function $[\![\cdot]\!]_{\mathrm{P}}$ computes likelihood estimations of decisions for a query based on the degree of similarity between attribute values. However, an access control mechanism can only enforce conclusive decisions, either *permit* or *deny*. Therefore, the IoT platform should resolve the uncertainty underlying the obtained likelihood estimations. Resolving such uncertainty might introduce risks of wrongly granting/denying access [8], which can result in breaches of sensitive information or can affect business continuity.

To assist the IoT platform in decision making, we employ a risk-based approach capable of quantifying the uncertainty inherent in likelihood estimations of decisions and reaching a conclusive decision. First, we note that the *not-applicable* decision cannot be directly enforced by the IoT platform. In this work, we adopt a conservative approach where the *not-applicable* decision is resolved into a *deny* decision. Moreover, the risk that can be tolerated by the IoT platform can vary from resource to resource. For instance, the waste of the goods caused by a high temperature in Use Case 5 can be deemed more critical than switching on/off the lights in Use Case 4.

This intuition can be formalized as follows. Given an access control policy p and a query q for a resource r such that $[\![q]\!]_{\mathrm{P}}(p) = (\ell^1, \ell^0, \ell^\perp)$, access to r is granted according to p if and only if:

$$\ell^1 \geq \alpha(\ell^0 + \ell^\perp), \tag{4}$$

where parameter $\alpha \geq 1$ is a *risk factor* representing the level of criticality of the resource according to the owner. In particular, Eq. 4 requires that the likelihood of obtaining a *permit* is at least equal to the likelihood of obtaining a *deny* or a *not-applicable* decision for non-critical resources (i.e., resources such as α is close to 1) and requires more evidence that the request should be permitted for more critical resources (i.e., resources such as $\alpha \gg 1$).

3.6 Protocol Details

Fig. 3 shows the protocol used by the client to request access to a service provided on the IoT platform (server). Note that we assume that the interactions between the client and the server occur over a secure Internet connection, adopting well-known secure connection protocols such as SSL/TLS or DTLS [6].

The protocol consists of the following steps:

(1) The client, via the PH, sends a request for a service r to the IoT platform. On message reception, the PH at the server side forwards the request to the RR, which in turn sends the request to the PPDP for evaluation.
(2) The PPDP evaluates the request against the policies fetched from the PAP (we do not explicitly represent the PAP in Fig. 3 and assume that policies are loaded in the PPDP at deployment time, in line with the XACML reference architecture [12]).
(3) If additional attributes are needed for policy evaluation, the PPDP requests the client to provide them through the PH. Note that, following to the XACML standards, these attributes can be determined either during policy evaluation when needed (represented by the inner loop in Fig. 3) or in advance. In Sec. 4, we evaluate and compare the computation costs and energy consumption of these approaches.
(4) The PH on the client forwards the request to the DRA, that evaluates the disclosure risk associated to the attributes requested by the IoT platform, (see Sec. 3.3). Specifically, the client determine the *non-sensitive* value for each requested attribute (cf. Def. 4).
(5) The attribute values are returned to the IoT platform via the PH. Based on the received values, the PPDP evaluates the likelihood

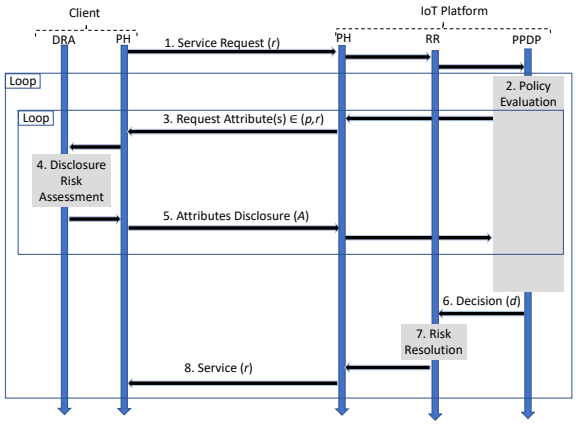

Figure 3: Sequence diagram of the protocol used by the client to access services provided by the server IoT platform.

that the decision is either *allow*, *deny*, or *not applicable*, namely $(\ell^1, \ell^0, \ell^\perp)$, as described in Sec. 3.4.

(6) The likelihood estimations $(\ell^1, \ell^0, \ell^\perp)$ are sent to the RR module for enforcement.

(7) The RR module evaluates the risks of providing access to r and generates a decision, either *allow* or *deny*, according to the rationale described in Sec. 3.5.

(8) If the decision is positive (i.e., *permit*), the RR grants the access to r to the client. Otherwise, if the decision is *deny*, the client is notified that he is not allowed to access r.

Note that the client can follow several approaches for the disclosure of the attribute values. For instance, the client could decide to immediately disclose the more specialized attribute values that satisfy her risk tolerance threshold or could opt for disclosing the most general attribute values in the first iteration of the protocol and then iteratively disclose the attribute values with higher disclosure risk at each step, thus disclosing every time additional amount of information compared to the previous round of the protocol. In particular, in case the final decision is *deny* (step 7 of Fig. 3), the client can decide to disclose a more specialized set of attributes, exposing more information to the server to access the service. Accordingly, the protocol continues in a loop as per the steps 4 to 8, until at least one of the two following conditions applies: *(i)* the access to the resource is granted, or *(ii)* the client cannot disclose any further value for a required attribute.

Although the latter approach minimizes the disclosure risk, it has two major drawbacks: *(i)* it increases the protocol delay, and *(ii)* it increases the overall energy consumption of both parties. Considering that the access to the service could be required to be in real-time and that the client could be energy-constrained, the client should suitably trade-off between the disclosure risk and the local time/energy requirements when adopting the presented protocol. We provide an in-depth evaluation of such a trade-off in Sec. 4.

4 PERFORMANCE ASSESSMENT

In this section, we report the details of our implementation (Sec. 4.1), experiment settings (Sec. 4.2), and results (Sec. 4.3).

4.1 Implementation Details

To quantify the trade-off between energy and privacy characterizing PICO, we implemented the framework in Matlab, plugging in experimental values from the real-world.

In our implementation, we embedded each disclosed attribute value into a dedicated signed object, namely a CBOR Web Token (CWT) [5]. A CWT is a compact tool to transfer information among parties, similarly to JSON Web Tokens (JWTs) in the context of traditional access control systems. CWTs consist of three parts. First, they include a header, reporting metadata used for correct decoding. Then, they include a set of claims, that can be either standardized or private (i.e., created ad-hoc), reporting several information about the CWT, e.g., its issuer, subject, and expiration date. Finally, they include a signature field, used for integrity and authenticity verification. CWTs are optimized for usage in IoT scenarios, where bandwidth and computational capabilities are usually scarce compared to traditional IT systems. Thus, the claims in a CWT are encoded using the Concise Binary Object Representation (CBOR) format, and they use CBOR Object Signing and Encryption (COSE) for additional application-layer security protection [20].

To implement CWTs, we used a dedicated python implementation (https://pypi.org/project/cwt/), integrating as standardized claims the issuer (*iss*), unique identifier (*cti*), subject (*sub*), and expiration date (*exp*). We also added an ad-hoc private claim for attribute values (*atv*). Finally, we used HMAC-SHA-256 to generate the signature of the CWT. As a result, a single CWT is 233 bytes.

To identify the number of MAC-layer messages needed to deliver a single CWT over an IoT wireless network, we adopted the protocol stack used by the IETF 6tisch WG and OpenWSN operating system [27], as described in [26]. Specifically, we assume the CWT is encapsulated into a CoAP message (4 bytes header), and then progressively into UDP (8 bytes header), 6LoWPAN (40 bytes header) and IEEE 802.15.4 (21 bytes header and 4 bytes trailer), to be delivered over the wireless channel by the IoT device. Since the Maximum Transmission Unit (MTU) of IEEE 802.15.4 is 127 bytes, 50 bytes are available in the payload for application-layer information. Thus, five IEEE 802.15.4 messages are needed to transfer an entire CWT.

To estimate the energy consumption of a reference IoT device, we used the experimental consumption data provided in [24]. This work used a state-of-the-art Openmote-b IoT device equipped with the IEEE 802.15.4 technology and the same protocol stack discussed above, measuring an average energy consumption of 802.65 mJ and 778.51 mJ for a transmission and a reception slot, respectively.

4.2 Experiment Settings

We performed two experiments dedicated to the evaluation of energy consumption and its trade-off with privacy guarantees associated to different operational strategies of our protocol. In particular, we identified two attribute disclosure approaches and two attribute delivery modes of the scheme discussed in Sec. 3.6. For attribute disclosure, we devised two approaches: *(A1)* direct disclosure of non-sensitive attribute values with the highest disclosure risk; *(A2)* incremental disclosure of non-sensitive attribute values, from the ones with the lowest disclosure risk to the ones with the highest disclosure risk, until the disclosed attribute values satisfy the policy employed by the server. For attribute delivery, we identified two

modes: *(M1)* all CWTs are delivered together in a single stream to the IoT platform; *(M2)* CWTs are delivered one-by-one in different streams to the IoT platform. As a result, we considered four operational strategies: (A1+M1), (A1+M2), (A2+M1), and (A2+M2).

In the first experiment, we evaluated the energy consumption and disclosure risk associated to the identified operational strategies. To this end, we selected a fixed number of attributes equal to 6, with a static configuration of the attribute hierarchy. For each attribute, we configured the hierarchy as a perfectly height-balanced binary tree with a predefined depth that ranges from 9 to 11. The semantic closeness between two directly-connected nodes is generated randomly but ensuring that the semantic closeness of sibling nodes with the parent node always sum up to 1. At the server side, we configured the server in a way to take the decision based on a random policy, encompassing a random number of attributes, from 1 to 6, and a random *risk factor*. Thus, for each new seed in the experiment, the IoT platform picks a random combination of the attributes to generate the policy and defines a random risk factor. The IoT device then returns a non-sensitive value for every requested attribute. The IoT platform grants access only if the set of attribute values provided by the IoT device satisfies Eq. 4.

In the second experiment, we evaluated the impact of the number of attributes to be disclosed on the battery consumption of the IoT device. This allows us to gain additional insights on the strategy that can be adopted by the IoT device w.r.t. the number of attributes to be disclosed. For the experiment, we kept the configuration of the first experiment mostly unchanged, only varying the number of attributes requested by the IoT platform from 1 to 10.

The experiments were performed using Matlab R2021a through simulations run on an HP ZBook Studio G5, equipped with two (2) Intel Core i7-9750H processors running at 2.60 GHz, 32 GB of RAM, and 1 TB of HDD storage.

4.3 Results

Fig. 4 reports the energy consumption of a reference Openmote-b IoT device in the settings of the first experiment, running the four identified strategies. Each box represents the distribution of energy consumption required by a strategy for 1,000 seeds. We can observe that the strategies based on the attribute disclosure approach (A1) are much more energy-efficient than the ones based on (A2). This occurs because (A1) requires only a single instance of the protocol described in Sec. 3.6. Conversely, with (A2), the client could need a number of interactions equal to the number of combinations of non-sensitive attribute values, where each combination comprises exactly one value for each requested attribute. In terms of MAC-layer frames (and delay), (A1) requires 17.89 IEEE 802.15.4 frames, while (A2) requires 48,512.58 frames (on average). We also notice that the mode (M1) is slightly more energy efficient than (M2); this is because chunks of CWTs can be aggregated into the same MAC-layer message, helping reducing the number of required messages and, thus, the resulting energy overhead. Considering the attribute disclosure approach (A1), (M1) and (M2) require 17.4 and 18.39 IEEE 802.15.4 frames, respectively.

We also evaluated the disclosure risk associated to the set of attribute values used by the IoT platform to take the final decision. The results are shown in Fig. 5; note that we only consider the

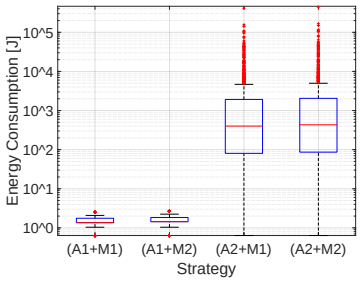

Figure 4: Energy consumption of a reference Openmote-b IoT device with each of the four strategies.

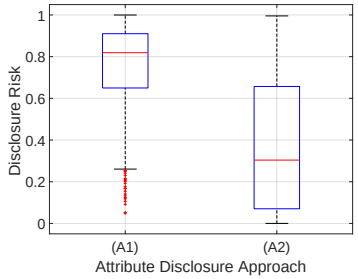

Figure 5: Disclosure Risk of the IoT device w.r.t. the attributes disclosure approaches.

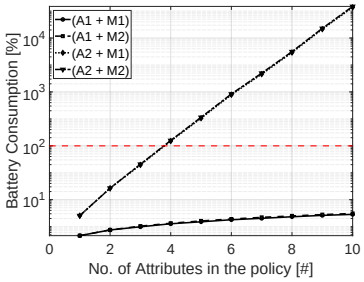

Figure 6: Battery consumption of a reference Openmote-b IoT device for each strategy at the vary of the number of attributes in the access control policy.

attribute disclosure approaches, as the attribute delivery modes have no effect on the disclosure risk. We can observe that, although being the most energy-efficient solution, (A1) is the least privacy-friendly (median value of disclosure risk is 0.82 vs. 0.3 for (A2)). Indeed, directly disclosing the non-sensitive attribute values with the highest disclosure risk discloses the most information about the actual (sensitive) values, offering more chances to the IoT platform to infer them. Conversely, (A2) allows the IoT device to minimize the information disclosed by providing attribute values at a higher level of granularity, thus resulting more privacy-preserving.

In the second experiment, we evaluated the impact of the number of attributes on the battery consumption of the IoT device for the four operational strategies. To obtain the percentage of battery consumption, we divided the energy consumption of each strategy by the overall battery capacity of the Openmote-b hardware board, available from [21]. The results are reported in Fig. 6. In line with the results of the previous experiment, we observe that the attribute disclosure approach *(A1)* is always more energy-efficient than *(A2)*,

especially when the number of attributes to be disclosed increases. In particular, the battery consumption for (A2) is exponential in the number of attributes to be disclosed. It is worth noting that it is not always possible for an Openmote-b board to use (A2) as the attribute disclosure approach to preserve its privacy. Indeed, when more than three attributes are needed, using (A2) this device consumes, on average, more than 100% of its battery capacity, exhausting all available energy to preserve its privacy in accessing just a single service. Nonetheless, it is worth noting that these results cannot be generalized to other types of IoT devices, and the number of attributes that can be supported could be larger for more powerful devices.

The above considerations suggest that an IoT device should carefully trade-off between privacy and energy consumption. For instance, the IoT device could decide to disclose only a subset of possible combinations of the attribute values, partly sacrificing its privacy to increase battery lifetime. Alternatively, it could decide to perform a few protocol rounds using the most privacy-preserving strategy, and then to either stop or use the most *risky* strategy, due to the excessive energy consumption derived by the high number of messages. Overall, PICO offers to IoT devices the unique opportunity to trade-off between privacy and energy consumption, offering enhanced situational awareness and the chance of choosing the attribute disclosure approach that best fits its energy availability and access delay requirements. This is a unique feature of PICO that, to the best of our knowledge, is not provided by any other contribution in the literature (cf. Sec. 5).

5 RELATED WORK

Several contributions in the last years focused on privacy-preserving access control in IoT scenarios.Table 2 provides an overview of existing solutions against the features identified in Sec. 2, cross-comparing them with PICO. Client privacy is typically achieved by decoupling attributes from the entity possessing them, through anonymization techniques. For instance, Ouaddah et al. [14] propose FairAccess, an access control solution using pseudonymous, preserving IoT devices' privacy by replacing the authority with a decentralized solution based on Blockchain. When applied in our context, the use of Blockchain technology results in a significant storage and communication overhead. Moreover, IoT devices should reveal their private values, enabling the system to derive correlations between their identity and attributes. In general, approaches based on the unlinkability between users and their attributes cannot fully guarantee anonymity, especially when the possession of specific attributes or attribute values leads to direct identification of the entity possessing it. Recently, ABE gained momentum as a general scheme to enforce access control without requiring users to expose their attributes. For instance, Nasiraee et al. [11] propose an ABE scheme preserving client privacy, even when multiple authorities collude, and evaluate its energy consumption on a smartphone. However, employing ABE on more constrained devices might not be possible, due to bandwidth and energy demands [15].

To cope with the demanding requirements of ABE on IoT devices, many proposals rely on trusted nodes, in line with emergent paradigms such as edge and fog computing. For instance, Fan et al. [4] present a variant of the Multi-Authority Ciphertext ABE (MA-ABE) cryptographic primitive [3] as a building block for a privacy-preserving outsourced multi-authority access control scheme, where attribute authentication and ciphertext decryption are outsourced to powerful fog nodes. Although reducing the computational and energy requirements compared to applying ABE locally on IoT devices, such scheme exposes sensitive attribute values to fog nodes, which might be malicious or compromised [23]. Similar considerations apply to the work in [19], which focuses on minimizing communication costs between IoT devices and the fog node, without considering the energy limitations of IoT devices. Taking into account that fog nodes might not be fully trusted, Xue et al. [29] propose a scheme that still relies on outsourcing, but uses two distinct fog nodes to carry out computations. However, the scheme is secure against ciphertext disclosure only under the assumption that such fog nodes do not collude.

Other solutions adapt the concept of anonymous credentials to IoT scenarios. For instance, Nasiraee et al. [10] propose a lightweight approach allowing users to access services anonymously, while protecting the access policy associated to such services. However, their method still outsources expensive computations to the Cloud, thus being not fully privacy-preserving for IoT devices. Similarly, Sanchez et al. [18] adapt Idemix on IoT devices, allowing a potentially constrained device to use anonymous credentials, and possibly not disclosing its attributes. Still, due to the computational constraints of devices, they use computation offloading techniques.

Server privacy and, in particular, the confidentiality of access control policies is addressed by Zhang et al. [30], which present a scheme preventing the guessing of the policy. Such a scheme also protects the attribute values by embedding them in a Bloom Filter. This approach cannot fully preserve client privacy, as an external entity with some degree of knowledge of sensitive attribute values can verify its possession. A proposal addressing both client and server privacy is the one by Xu et al. [28], proposing to use Homomorphic Encryption (HE) to hide attribute values while allowing the enforcement of access control policies. However, HE is typically too computationally and bandwidth expensive to be run on constrained IoT devices. PICO is based on the work in [16], which proposes a risk-based access control framework handling incomplete information in policy evaluation, leveraging the semantic closeness between attributes. In our work, we extend the framework in [16] by including privacy-related considerations and contextualizing it in IoT use cases. In particular, we enable constrained IoT devices to evaluate the risk associated to the disclosure of specific attributes values and to disclose them in a way to trade-off between privacy and energy constraints. Also other approaches propose strategies for risk-based access control in IoT scenarios based on the past behavior of the requester [2] or relying on adaptive monitoring based on game theory and context awareness [1]. However, they are not applicable to our context, as they cannot quantify the risks associated to the use of coarse-grained information for policy evaluation.

To the best of our knowledge, PICO is the only solution allowing IoT devices to access resources while not disclosing sensitive attributes values, not relying on computationally expensive techniques. Moreover, PICO does not require outsourcing to trusted entities, while allowing for the local assessment of the disclosure risk of any attribute value. PICO also enables IoT devices to trade-off between energy consumption and privacy, choosing the attribute disclosure strategy that best fits their computational/energy budget

Table 2: Qualitative comparison of privacy-preserving access control solutions for IoT.

In the table, ● means "*full support*", ◐ "*partial support*", ○ "*no support*". We use N/A to indicate that a certain requirement is *not applicable* to a certain solution. For instance, solutions that always disclose the exact attribute value do not need to provide mechanism for **Server Risk Assessment** as the server does not take any risk of inadvertently granting (deny) access to unauthorized (authorized) users.

	Client Privacy				Server Privacy	Energy	
	No disclosure sensitive info.	**Disclosure Risk (Client)**	**Server Risk Assessment**	**No computations outsourcing**	**Policy Confidentiality**	**Evaluation**	**Privacy Trade-off**
Ouaddah et al. [14]	○	○	N/A	●	○	○	○
Nasiraee et al. [11]	●	○	N/A	●	●	●	○
Fan et al. [4]	●	○	N/A	○	●	○	○
Sarma et al. [19]	●	○	N/A	○	○	○	○
Xue et al. [29]	●	○	N/A	○	○	○	○
Nasiraee et al. [10]	●	○	N/A	○	●	○	○
Sanchez et al. [18]	●	○	N/A	○	○	○	○
Zhang et al. [30]	○	○	N/A	●	●	○	○
Xu et al. [28]	●	○	N/A	○	●	○	○
PICO	●	●	●	●	◐	●	●

and privacy requirements. Although being not our primary objective, PICO also does not expose access control policies to the client, thus partially preserving their confidentiality.

6 CONCLUSIONS AND FUTURE WORK

This work presented PICO, a framework enabling privacy-preserving access control in IoT scenarios through incomplete information. In a nutshell, PICO allows IoT devices to disclose coarse-grained attribute values, based on an attribute-based disclosure risk. PICO also provides a mechanism for IoT platforms to evaluate access policies using such incomplete information, and to possibly provide access to services by quantifying the risks associated with granting or denying access. Our results show that PICO can provide different levels of privacy, with the most energy-efficient strategy (battery consumption below 1%) being also the least privacy-preserving, and allows IoT devices to achieve more privacy at the cost of energy consumption. Thus, IoT devices running PICO can trade-off between privacy and energy on a case-by-case basis.

Future work include extending PICO to manage missing attributes and deploying a proof-of-concept in a complex IoT scenario.

ACKNOWLEDGEMENTS

This work has been partially supported by the INTERSCT project, Grant No. NWA.1162.18.301, funded by Netherlands Organisation for Scientific Research (NWO). The findings reported herein are solely responsibility of the authors.

REFERENCES

[1] H. Abie and I. Balasingham. 2012. Risk-based adaptive security for smart IoT in eHealth. In *BODYNETS*. 269–275.
[2] H. Atlam et al. 2017. Developing an Adaptive Risk-Based Access Control Model for the Internet of Things. In *iThings*. IEEE, 655–661.
[3] M. Chase. 2007. Multi-Authority Attribute Based Encryption. In *Theory of Cryptography*. Springer, 515–534.
[4] K. Fan et al. 2019. Efficient and privacy preserving access control scheme for fog-enabled IoT. *Future Gener. Comput. Syst.* 99 (2019), 134–142.
[5] M. Jones et al. 2018. CBOR Web Token (CWT). *RFC 8392, IETF* (2018).
[6] T. Kothmayr et al. 2013. DTLS based security and two-way authentication for the Internet of Things. *Ad Hoc Networks* 11, 8 (2013), 2710–2723.
[7] P. Leu et al. 2018. I Send, Therefore I Leak: Information Leakage in Low-Power Wide Area Networks. In *WiSec*. ACM, 23–33.
[8] I. Molloy et al. 2012. Risk-based security decisions under uncertainty. In *CODASPY*. ACM, 157–168.
[9] C. Morisset and N. Zannone. 2014. Reduction of access control decisions. In *SACMAT*. ACM, 53–62.
[10] H. Nasiraee and M. Ashouri-Talouki. 2020. Anonymous decentralized attribute-based access control for cloud-assisted IoT. *Fut. Gener. Comp. Syst.* (2020), 45–56.
[11] H. Nasiraee et al. 2021. Privacy-Preserving Distributed Data Access Control for CloudIoT. *TDSC* (2021), 1–1.
[12] OASIS. 2013. *eXtensible Access Control Markup Language (XACML) Version 3.0*. OASIS Standard. OASIS.
[13] G. Oligeri, S. Sciancalepore, S. Raponi, and R. Di Pietro. 2020. BrokenStrokes: on the (in) security of wireless keyboards. In *WiSec*. ACM, 231–241.
[14] A. Ouaddah et al. 2017. Towards a novel privacy-preserving access control model based on blockchain technology in IoT. In *Advances in information and communication technologies*. Springer, 523–533.
[15] P. Perazzo et al. 2021. Performance evaluation of Attribute-Based Encryption on constrained IoT devices. *Comput. Commun.* 170 (2021), 151–163.
[16] S. Ravidas, I. Ray, and N. Zannone. 2020. Handling Incomplete Information in Policy Evaluation using Attribute Similarity. In *TPS-ISA*. IEEE, 79–88.
[17] Reuters. 2021. Morgan Stanley faces data breach, corporate client info stolen in vendor hack. https://tinyurl.com/vnssavh9. (Accessed: 2021-10-04).
[18] J. Sanchez, J. Bernabe, and A. Skarmeta. 2018. Integration of anonymous credential systems in IoT constrained environments. *IEEE Access* 6 (2018), 4767–4778.
[19] R. Sarma et al. 2021. PAC-FIT: An efficient privacy preserving access control scheme for fog-enabled IoT. *Sustain. Comput.: Informa. and Sys.* 30 (2021), 100527.
[20] J. Schaad. 2017. CBOR Object Signing and Encryption. *RFC 8152, IETF* (2017).
[21] Savio Sciancalepore and Roberto Di Pietro. 2019. Bittransfer: Mitigating Reactive Jamming in Electronic Warfare Scenarios. *IEEE Access* 7 (2019), 156175–156190.
[22] M. Sheikhalishahi, G. Tillem, Z. Erkin, and N. Zannone. 2019. Privacy-Preserving Multi-Party Access Control. In *WPES*. ACM, 1–13.
[23] P. Tedeschi et al. 2019. Edge and fog computing in critical infrastructures: Analysis, security threats, and research challenges. In *EuroS&PW*. 1–10.
[24] P. Tedeschi et al. 2019. LiKe: Lightweight certificateless key agreement for secure IoT communications. *IEEE Internet of Things J.* 7, 1 (2019), 621–638.
[25] The Verge. 2021. T-Mobile data breach exposed the personal info of more than 47 million people. https://tinyurl.com/hjfpaz5p. (Accessed: 2021-10-04).
[26] X. Vilajosana et al. 2019. IETF 6TiSCH: A Tutorial. *IEEE Commun. Surv. Tutor.* 22, 1 (2019), 595–615.
[27] T. Watteyne et al. 2012. OpenWSN: a standards-based low-power wireless development environment. *Trans. Emerg. Telecommun. Technol.* 23, 5 (2012), 480–493.
[28] Y. Xu et al. 2018. A Privacy-Preserving Attribute-Based Access Control Scheme. In *SpaCCS*. Springer, 361–370.
[29] K. Xue et al. 2018. Fog-Aided Verifiable Privacy Preserving Access Control for Latency-Sensitive Data Sharing in Vehicular Cloud Computing. *IEEE Network* 32, 3 (2018), 7–13.
[30] L. Zhang, J. Wang, and Y. Mu. 2021. Privacy-preserving Flexible Access Control for Encrypted Data in Internet of Things. *IEEE Internet of Things J.* (2021), 1–1.