symmetry

MDPI

Article

An Effective Naming Heterogeneity Resolution for XACML Policy Evaluation in a Distributed Environment

Teo Poh Kuang [1], Hamidah Ibrahim [1,*], Fatimah Sidi [1], Nur Izura Udzir [1] and Ali A. Alwan [2]

[1] Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang 43400, Malaysia; gs23442@student.upm.edu.my (T.P.K.); fatimah@upm.edu.my (F.S.); izura@upm.edu.my (N.I.U.)

[2] School of Theoretical & Applied Science, Ramapo College of New Jersey, Mahwah, NJ 07430, USA; aaljuboo@ramapo.edu

* Correspondence: hamidah.ibrahim@upm.edu.my

Abstract: Policy evaluation is a process to determine whether a request submitted by a user satisfies the access control policies defined by an organization. Naming heterogeneity between the attribute values of a request and a policy is common due to syntactic variations and terminological variations, particularly among organizations of a distributed environment. Existing policy evaluation engines employ a simple string equal matching function in evaluating the similarity between the attribute values of a request and a policy, which are inaccurate, since only exact match is considered similar. This work proposes several matching functions which are not limited to the string equal matching function that aim to resolve various types of naming heterogeneity. Our proposed solution is also capable of supporting symmetrical architecture applications, in which the organization can negotiate with the users for the release of their resources and properties that raise privacy concerns. The effectiveness of the proposed matching functions on real XACML policies, designed for universities, conference management, and the health care domain, is evaluated. The results show that the proposed solution has successfully achieved higher percentages of Recall and F-measure compared with the standard Sun's XACML implementation, with our improvement, these measures gained up to 70% and 57%, respectively.

Keywords: access control policies; policy evaluation; naming heterogeneity; XACML

check for updates

Citation: Kuang, T.P.; Ibrahim, H.; Sidi, F.; Udzir, N.I.; Alwan, A.A. An Effective Naming Heterogeneity Resolution for XACML Policy Evaluation in a Distributed Environment. *Symmetry* **2021**, *13*, 2394. https://doi.org/10.3390/sym13122394

Academic Editor: Tomohiro Inagaki

Received: 16 November 2021
Accepted: 7 December 2021
Published: 12 December 2021

1. Introduction

Policy evaluation is a process to determine whether a request submitted by a user satisfies the access control policies defined by an organization. A practical distributed policy evaluation framework should be able to support the autonomy in policy specification, as well as interoperability, among parties and policy portability [1–5]. Naming heterogeneity arises due to the use of different combinations of characters which can represent the same term (syntactic variations), including typographical errors, similar terms belonging to different grammar categories, and different terms which have the same meaning (terminological variations) [6,7].

Existing policy evaluation engines [8–11] employ a simple string equal matching function during policy evaluation. However, they are deemed inaccurate since they do not explore naming heterogeneity and rely on the assumption that different terms represent different concepts. It would be unrealistic to assume that different organizations from different security domains would share the same vocabulary to represent their policies.

Several researchers have used ontologies for the specification of policies or add on semantic knowledge-based functions for semantic interoperability [12–19]. However, ontology forming is a labor-intensive, error-prone, and time-consuming task because, in general, it involves human input during the policy design stage to manually perform the ontology concept mapping, and with an assumption that the security officer is trusted to perform an accurate mapping. Moreover, the ontology needs to be reformed once a new party joins the collaboration. Therefore, developing a matching function that attempts to achieve effectiveness has been one of the main tasks in policy evaluation.

Several matching functions are proposed in this work to resolve the issue of naming heterogeneity between the attribute values of a request and a policy during policy evaluation. The proposed solution is domain-independent as it does not rely on any specific rules of a particular domain, hence a predefined knowledge of the domain is not required. Tokenization and concatenation are applied to the attribute values in order to remove unnecessary delimiters, which are considered as noise, before the proposed matching functions are executed. *N-gram* and WordNet are adopted as well in the proposed solution. *N-gram* is effective in matching terms with minor syntactic differences [13]; while WordNet could identify the equivalence and inheritance relationships between the attribute values of a request and a policy.

This work is based on the discretionary access control (DAC) model. The eXtensible access control markup language (XACML) is used to specify the policy since it is the OASIS standard language, and the standard defines a declarative access control policy language implemented in XML format, which is able to express policies in terms of rules over different kinds of attributes. Overall, the main contributions of this work are briefly described as follows:

- We have proposed a naming heterogeneity resolution model with the main aim to resolve naming heterogeneity, which may arise due to syntactic variations and terminological variations during policy evaluation.
- Several matching functions have been proposed. Each matching function has been designed to cater to certain type of variation (syntactic and/or terminological) by analyzing the terms that appeared in the attribute values of a request and a policy. *N-gram* and WordNet are utilized to provide the syntactic similarities and semantic relationships (synonym, hypernym, and hyponym) between terms, respectively.
- The experimental results of the proposed solution are presented to prove its capability of identifying and resolving naming heterogeneity due to syntactic and terminological variations during policy evaluation.

The rest of the paper is organized as follows. Section 2 reviews the methods of policy evaluation proposed by previous studies. Section 3 introduces the necessary definitions and notations used throughout the paper, while Section 4 presents the proposed matching functions that aim to resolve naming heterogeneity, namely: *Synonym Equal*, *Hyponym*, *Syntactical Synonym Equal*, *Syntactical Hyponym*, *Syntactical Equal*, *Hyponym Common Word*, and *Abbreviation Equal*. An illustrative example based on the academy university domain is also given. Section 5 evaluates the performance of the proposed matching functions which is then compared to the performance of a previous notable work. The last section concludes this work and sheds light on some directions which can be used in the future.

2. Related Works

Numerous studies have proposed methods for integrating policies of collaborating parties into a global policy schema, which may support complex authorization specifications and requirements of the collaborating parties [19–26]. However, policy integration methods among various collaborating parties could become very complex due to domain heterogeneity and different vocabulary utilized by organizations in specifying their policies.

Several works have affirmed that collaborative partners may need to perform policy similarity by comparing their access control policies in order to determine which requests will be permitted among the policies [27–30]. Nevertheless, these works required the collaborative parties to provide their individual and independent policies that may be misused by adversaries with the intention to reveal sensitive information among those policies and may lead to unintended breaches of privacy. Due to the difficulty of integrating schemas from different organizations into a global schema, current researchers providing solutions for dynamic policy evaluation, which fit in the large scale of distributed systems, are receiving particular attention [8–11].

Sun's XACML implementation [11] is a policy evaluation mechanism that is specifically designed to provide a full support for determining applicability of policies and evaluating requests against policies in XACML. The major problems are that XACML is unable to handle semantics that are associated with the elements and unable to properly detect policy conflicts among complex policies [5].

In order to achieve an efficient XACML policy evaluation that is able to deal with a large volume of requests, several works have focused on the performance of processing requests by improving the Sun's XACML implementation [8–10]. These works, which mainly focus on evaluation time, have adopted a simple string equal matching function to match the string values. However, the simple string-based method is unable to solve naming heterogeneity in a distributed environment since we cannot expect that policies belonging to different organizations are based on the same vocabulary. Nevertheless, there are also works such as [2,31–40] that made attempts to improve the policy decision point (PDP) evaluation performance with regard to evaluation time by grouping/clustering the whole rule set into several subsets; hence, resolving the issue of semantic interoperability is not part of their solutions. Meanwhile, our work focuses on the effectiveness of a policy evaluation engine in which accuracy is the main measurement used.

A number of works have supported semantic interoperability to resolve naming heterogeneity [4,7,12–15,17,18]. These works combined policy rules with ontologies in order to improve the query answering support to infer domain knowledge. However, the ontology-based knowledge management in these works is a labor-intensive, error-prone, and time-consuming task because it needs intensive human involvement during the access control policy design stage to manually map the ontology. The human perception error that might occur while performing mapping, especially for policies of larger sizes, further hinders the full acceptance of such solution.

All possible violations that might exist among a request and a policy are identified based on the subject, object, action, and condition attributes of the request and policy. However, existing works are still lacking in terms of providing solutions to resolve naming heterogeneity, and it is yet to be validated whether the results returned by the evaluation engines are accurate. According to the work in [41], to reduce human involvement, string-based and language-based techniques and linguistic resources can be used to analyze strings. In [42], the authors measure syntactical similarity by using *N-gram* to compute the number of common *N-grams* between terms (i.e., sequences of *N* characters) while terminological analysis is performed by identifying equality between concepts utilizing the WordNet lexical database.

This work is an extension to our previous work [42], in which WordNet is further utilized to identify equivalence and inheritance relationships between the attribute values of a request and a policy. Table 1 presents a summary of the existing naming heterogeneity methods in a distributed system as described in this section.

Table 1. The summary of the existing naming heterogeneity methods in a distributed system.

A	*I*	*II*					*III*	
	B	***C***	***D***	***E***	***F***	***G***	***H***	***I***
[23]	√	-	String Equal	Graph Matching	String Equal	String Equal	√	√
[26]	√	-	Vector similarity, clustering, ontology graph matching	Vector similarity, clustering, ontology graph matching	String Equal	-	√	√

Table 1. *Cont.*

A	*I*		*II*				*III*	
	B	***C***	***D***	***E***	***F***	***G***	***H***	***I***
[24]	√	-	Equal, not equal, intersect, subset, superset	Equal, not equal, intersect, subset, superset	Equal, not equal, intersect, subset, superset	Equal, not equal, intersect, subset, superset	√	-
[25]	√	-	Addition, subtraction, negation, domain projection	Addition, subtraction, negation, domain projection	Addition, subtraction, negation, Domain, Projection	Addition, subtraction, negation, domain projection	√	-
[41]	√	-	Addition, subtraction, intersection, precedence, negation, domain projection	Addition, subtraction, intersection, precedence, negation, domain projection	Addition, subtraction, intersection, precedence, negation, domain projection	Addition, subtraction, intersection, precedence, negation, domain projection	√	-
[20]	√	-	Intersection, scoping restriction, set difference	Intersection, scoping restriction, set difference	Intersection, scoping restriction, set difference	Intersection, scoping restriction, set difference	√	-
[21]	√	-	String Equal	String Equal	String Equal	Algebraic	√	-
[22]	√	-	Ontology graph matching	Ontology graph matching	Ontology graph matching	Ontology graph matching	√	√
[30]	√	-	Ontology graph matching	Ontology graph matching	Ontology graph matching	Ontology graph matching	√	-
[29]	√	-	Domain specific thesauri, WordNet, ontology graph matching	Domain specific thesauri, WordNet, ontology graph matching	Domain specific thesauri, WordNet, ontology graph matching	Domain specific thesauri, WordNet, ontology graph matching	√	√
[27]	√	-	Domain specific thesauri, WordNet, ontology graph matching	Domain specific thesauri, WordNet, ontology graph matching	Domain specific thesauri, WordNet, ontology graph matching	Domain specific thesauri, WordNet, ontology graph matching	√	√
[11]	-	√	String Equal	String Equal	String Equal	String Equal	√	-
[18]	√	-	Jena and Pellet reasoner	Jena and Pellet reasoner	Jena & Pellet reasoner	Jena & Pellet reasoner	√	√
[8]	-	√	String Equal	String Equal	String Equal	String Equal	√	-
[9]	-	√	String Equal	String Equal	String Equal	String Equal	√	-
[10]	-	√	String Equal	String Equal	String Equal	String Equal	√	-
[12]	-	√	JaroWinklerTF-IDF, WordNet, user dictionary, ontology graph matching	JaroWinklerTF-IDF, WordNet, user dictionary, ontology graph matching	JaroWinklerTF-IDF, WordNet, user dictionary, ontology graph matching	-	√	√
[14]	-	√	Ontology graph matching	Ontology graph Matching	Ontology graph matching	-	√	√
[13]	-	√	Pellet reasoner	Pellet reasoner	Pellet Reasoner	Pellet reasoner	√	√

Table 1. *Cont.*

A	*I*		*II*				*III*	
	B	***C***	***D***	***E***	***F***	***G***	***H***	***I***
[15]	-	√	Ontology graph matching	Ontology graph Matching	Ontology graph matching	-	√	√
[17]	-	√	Jena & Pellet reasoner	Jena & Pellet reasoner	Jena & Pellet Reasoner	Jena & Pellet Reasoner	√	√
[4]	-	√	Jena reasoner	Jena reasoner	Jena reasoner	Jena Reasoner	√	√
[7]	-	√	Ontology graph matching	Ontology graph matching	Ontology graph matching	Ontology graph matching	√	√
[42]	-	√	WordNet, *N-gram*	WordNet, *N-gram*	WordNet, *N-gram*	WordNet, *N-gram*	√	√

Note: *I*—environment; *II*—matching methods; *III*—variations; *A*—authors; *B*—matching two policies; *C*—matching a request and a policy; *D*—subject; *E*—resource; *F*—action; *G*—condition; *H*—syntactic; *I*—terminological.

3. Preliminaries

In this section, we present the necessary definitions and introduce the notations that are used throughout this paper. First, we give the general definitions related to policy evaluation that have been defined either formally or informally in the literature [6,7,15,43–45], based on the notations used in this paper (i.e., Definition 1 and Definition 2). This is then followed with specific definitions that are related to our work. Motivation examples are then put forward to further clarify the problem addressed in this paper.

3.1. Definitions and Notations

The definitions of policy and request are as follows:

Definition 1. *An access control policy, Pol, is a tuple of the form: Pol* $\equiv$ *(Effect, Target, Condition).*

Definition 2. *A request, Req, is presented in the form: Req* $\equiv$ *(Subject, Resource, Action, Condition).*

A *Target* is basically a set of conditions of a subject, resource, and action that must be met for a policy to be applied to a given request. *Subject*, *Resource*, and *Action* are the components of a request and a policy. However, *Condition* is an optional attribute to further constrain the scope of a request or a policy. *Subject* identifies an individual user or a user role that can potentially invoke an action in the system. *Resource* can be any objects for the subject to access (e.g., data or computer resources such as Webservers or database servers). *Action* represents any operations (e.g., delete or write a file) that can be applied to the resource. Finally, *Condition* is a Boolean expression that involves environment context of evaluation. Examples of a typical environment context are time (e.g., 2 p.m. $\leq$ time $\leq$ 5 p.m.) and spatial (e.g., $location = Faculty\ Floor$). The effect is the intended consequence of a satisfied policy (either *Permit* or *Deny*).

We use the symbols $\vee$ as a logical disjunction and $\wedge$ as a logical conjunction when multiple terms are joined into a single attribute of an access control policy. Our work covers the domain elementary expressions which are classified into three categories [7], as follows:

Category 1. *One variable equality constraints,* $x = c$, *where x is a variable and c is a constant.*

Category 2. *One variable inequality constraints,* $x \rhd c$, *where x is a variable, c is a constant, and* $\rhd \in \{<, \leq, \geq, >\}$.

Category 3. *Compound Boolean expression constraints. This category combines the categories 1 and 2 using the logical operators* $\wedge$ *or* $\vee$.

The domain of the terms that appeared in the above constraints belongs to string data type (e.g., $Email = gs23442@upm.edu.my$) and date/time data type (e.g., $Time = 12:30$).

A policy is said to be applicable to a request if the term of a subject, resource, action, and condition of the request corresponds to the term of a subject, resource, action, and condition of the policy, respectively. The definition of an applicable policy can be formally defined as follows:

Definition 3. *A policy, Pol_i, is said to be applicable to a request, Req_j, if and only if the subject of the request, $Subject_{Req_j}$, corresponds to the subject of the policy, $Subject_{Pol_i}$, the resource of the request, $Resource_{Req_j}$, corresponds to the request of the policy, $Resource_{Pol_i}$, the action of the request, $Action_{Req_j}$, corresponds to the action of the policy, $Action_{Pol_i}$, and the condition of the request, $Condition_{Req_j}$, corresponds to the condition of the policy, $Condition_{Pol_i}$.*

Definition 4. *A term of Req_j, av_{Req_j}, is said to correspond to a term of Pol_i, av_{Pol_i}, if and only if:*

$$\left[\left(av_{Req_j} = av_{Pol_i}\right) \vee \left(av_{Req_j} \equiv av_{Pol_i}\right)\right] \wedge SC\left(av_{Req_j}, av_{Pol_i}\right) \geq \tau,$$

where $\equiv$ is an equivalence symbol, SC is a similarity score, and τ is a similarity threshold. Here, term implies the explicit value of an attribute of a request and a policy as specified by the user and administrator, respectively.

Definition 5. *The semantic relationship between a term of Req_j, av_{Req_j}, and a term of Pol_i, av_{Pol_i}, can be one of the following: synonym, hyponym, and hypernym. Synonym is a relation that exists between av_{Req_j} and av_{Pol_i} that have the same meaning. Hyponym is a relation between av_{Req_j} and av_{Pol_i} that implies one of terms is a specific meaning than the other term which is the general or superordinate term. The opposite relationship of hyponym is hypernym. Other relationships like antonym, homonym, and polysemy are not considered in this work since antonym represents a term opposite in meaning to another, homonym means that the terms are having the same spelling or pronunciation but different meanings and origins, and polysemy is the coexistence of many possible meanings for a term.*

3.2. Illustrative Example

This section presents an illustrative example, based on the academy university domain. It attempts to highlight the following: (i) the various forms of terms used in a request as well as a policy and (ii) the different types of naming heterogeneity that occur during policy evaluation. These variants of terms and heterogeneity further hinder the process of matching and evaluating the similarities between the attribute values of a request and a policy during policy evaluation. Table 2 presents five explicit access control policies, based on Definition 1, while Table 3 presents four requests, based on Definition 2.

Table 2. The XACML policies applied in the University.

Policy No.	Effect	Subject	Resource	Action	Condition
Pol_1	*Permit*	*RA*	*Grades*	$Assign \vee View$	$(Location = Association) \wedge (Time \geq 12$ p.m. $\wedge$ $Time \leq 2$ p.m.$) \wedge (Email = upm.edu.my)$
Pol_2	*Deny*	*Student*	*Course*	$Assign \vee View$	$(Location = Department) \wedge (Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.$) \wedge (Email = upm.edu.my)$
Pol_3	*Permit*	*Undergrad*	*Course*	*View*	$(Location = Department) \wedge (Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.$) \wedge (Email = upm.edu.my)$
Pol_4	*Permit*	$AssociateProfessor$	*Grades*	$SubmitGradeChange \vee SubmitGrade \vee Assign \vee View$	$(Location = GraduateSchool) \wedge (Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.$) \wedge (Email = upm.edu.my)$
Pol_5	*Deny*	*Faculty_Member*	*Grades*	Assign ∨ View	$(Location = School) \wedge (Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.$)$

Table 3. The requests for policy evaluation.

Request No.	Subject	Resource	Action	Condition
Req_1	*Undergraduate Student*	*Teaching Course*	*View*	$(Location = University\ Department) \wedge$ $(Time = 12:30$ p.m. $) \wedge$ $(Email = gs23442@upm.edu.my)$
Req_2	*ResearchAssistant*	*ExternalGrades*	*Assign*	$(Location = Institute) \wedge$ $(Time = 1:30$ p.m. $) \wedge$ $(Email = gs23442@upm.edu.my)$
Req_3	*AssociateProf*	*Grades*	*Assign*	$(Location = GraduateSchool) \wedge$ $(Time = 12:30$ p.m.$)$
Req_4	*Faculty_Member*	*Grades*	*AssignGrade*	$(Location = School) \wedge$ $(Time = 12:30$ p.m.$)$

Altogether, there are 20 comparisons (5 × 4) for this illustrative example but only those comparisons that imply the policies are applicable to the requests are shown in Tables 4–10. From this example, it is found that different forms of terms are used in a request and a policy, as further elaborated below:

- A compound noun is a noun that is made of two or more words. For instance, referring to Table 3, the term *Teaching Course* in the resource attribute of Req_1 is a compound noun.
- An *abbreviation* is a shortened or contracted form of a word or phrase. For instance, referring to Table 8, the term *Prof* which is part of the term *AssociateProf* in the subject attribute of Req_3 is a shortened form of *Professor*.
- An acronym is a word formed as an abbreviation from the initial letters in a phrase or a word [43]. For instance, referring to Table 2, *RA* in the subject attribute of Pol_1 is formed from the initial letters of *ResearchAssistant* in the subject attribute of Req_2 (Table 3).
- A word may appear at the beginning of another word which is in the form of a compound noun. For instance, referring to Table 9, *Assign* in the action attribute of Pol_4 occurs at the beginning of *AssignGrade* in the action attribute of Req_4.
- A word may appear at the end of another word which is in the form of a compound noun. For instance, referring to Table 6, *Grades* in the resource attribute of Pol_1 occurs at the end of *ExternalGrades* in the resource attribute of Req_2.
- A word may contain delimiter characters (i.e., dash, underscore, capital letters, etc.). For example, referring to Table 2, the term *Faculty_Member* in the subject attribute of Pol_5 contains "_" as delimiter.

Table 4. The mapping results among Req_1 and Pol_2.

Req_1	Pol_2	Result	Type of Variations
$Subject = Undergraduate\ Student$	$Subject = Student$	Match	Terminological
$Resource = Teaching\ Course$	$Resource = Course$	Match	Terminological
$Action = View$	$Action = Assign$	Not Match	-
	$Action = View$	Match	Syntactic
$Location = University\ Department$	$Location = Department$	Match	Terminological
$Time = 12:30$ p.m.	$Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.	Match	-
$Email = gs23442@upm.edu.my$	$Email = upm.edu.my$	Match	Terminological

Table 5. The mapping results among Req_1 and Pol_3.

Req_1	Pol_3	**Result**	**Type of Variations**
$Subject = Undergraduate\ Student$	$Subject = Undergrad$	Match	Syntactic
$Resource = Teaching\ Course$	$Resource = Course$	Match	Terminological
$Action = View$	$Action = View$	Match	Syntactic
$Location = University\ Department$	$Location = Department$	Match	Terminological
$Time = 12:30$ p.m.	$Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.	Match	-
$Email = gs23442@upm.edu.my$	$Email = upm.edu.my$	Match	Terminological

Table 6. The mapping results among Req_2 and Pol_1.

Req_2	Pol_1	**Result**	**Type of Variations**
$Subject = ResearchAssistant$	$Subject = RA$	Match	Syntactic
$Resource = ExternalGrades$	$Resource = Grades$	Match	Syntactic
$Action = Assign$	$Action = Assign$	Match	Syntactic
	$Action = View$	Not Match	-
$Location = Institute$	$Location = Association$	Match	Terminological
$Time = 1:30$ p.m.	$Time \geq 12$ p.m. $\wedge$ $Time \leq 2$ p.m.	Match	-
$Email = gs23442@upm.edu.my$	$Email = upm.edu.my$	Match	Terminological

Table 7. The mapping results among Req_3 and Pol_4.

Req_3	Pol_4	**Result**	**Type of Variations**
$Subject = AssociateProf$	$Subject = AssociateProfessor$	Match	Syntactic
$Resource = Grades$	$Resource = Grades$	Match	Syntactic
$Action = Assign$	$Action = Assign$	Match	Syntactic
	$Action = View$	Not Match	-
	$Action = SubmitGrade$	Not Match	-
	$Action = SubmitGradeChange$	Not Match	-
$Location = GraduateSchool$	$Location = GraduateSchool$	Match	Syntactic
$Time = 12:30$ p.m.	$Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.	Match	-

Table 8. The mapping results among Req_3 and Pol_5.

Req_3	Pol_5	**Result**	**Type of Variations**
$Subject = AssociateProf$	$Subject = Faculty\_Member$	Match	Terminological
$Resource = Grades$	$Resource = Grades$	Match	Syntactic
$Action = Assign$	$Action = Assign$	Match	Syntactic
	$Action = View$	Not Match	-
$Location = GraduateSchool$	$Location = School$	Match	Terminological
$Time = 12:30$ p.m.	$Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.	Match	-

Table 9. The mapping results among Req_4 and Pol_4.

$\boldsymbol{Req_4}$	$\boldsymbol{Pol_4}$	**Result**	**Type of Variations**
$Subject = Faculty\_Member$	$Subject = AssociateProfessor$	Match	Terminological
$Resource = Grades$	$Resource = Grades$	Match	Syntactic
$Action = AssignGrade$	$Action = Assign$	Match	Terminological
	$Action = View$	Not Match	-
	$Action = SubmitGrade$	Not Match	-
	$Action = SubmitGradeChange$	Not Match	-
$Location = School$	$Location = GraduateSchool$	Match	Terminological
$Time = 12:30$ p.m.	$Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.	Match	-

Table 10. The mapping results among Req_4 and Pol_5.

$\boldsymbol{Req_4}$	$\boldsymbol{Pol_5}$	**Result**	**Type of Variations**
$Subject = Faculty\_Member$	$Subject = Faculty\_Member$	Match	Syntactic
$Resource = Grades$	$Resource = Grades$	Match	Syntactic
$Action = AssignGrade$	$Action = Assign$	Match	Terminological
	$Action = View$	Not Match	-
$Location = School$	$Location = School$	Match	Syntactic
$Time = 12:30$ p.m.	$Time \geq 12$ p.m. $\wedge$ $Time \leq 1$ p.m.	Match	-

The different forms of terms cause naming heterogeneity between a request and a policy. Based on the illustrative example, two types of naming heterogeneity among terms that need to be addressed during policy evaluation are identified, which are *syntactic* and *terminological*. It is crucial to be able to recognize the form of the terms so as to achieve the appropriate matching functions in resolving the naming heterogeneity. String-based techniques (i.e., *N-gram*, prefix, and suffix), language-based techniques (i.e., tokenization), and linguistic resources [41] (i.e., WordNet) are the techniques that are suitable to be applied in resolving naming heterogeneity automatically, because of its ability to reduce human involvement.

4. Naming Heterogeneity Resolution

Figure 1 shows the general process flow of our proposed naming heterogeneity resolution model that aims to resolve naming heterogeneity that might occur during policy evaluation. It is possible that the policies from the resource organization do not directly match with the request since the terms used in the subject, resource, action, and condition are different. This is because each organization manages its own vocabulary of policies in order to serve their own authority's principal concern. Hence, naming heterogeneity is one of the heterogeneity issues that should be addressed in policy evaluation since the policies belonging to different organizations are not based on the same vocabulary.

A user may send a request to access the resources of an organization. The conflict resolution algorithm in this work compares the terms of a request, Req, against the terms of a policy, Pol. WordNet is applied as an external thesaurus with the purpose of identifying synonym, hypernym, or hyponym relationships between terms. The term may be vague in meaning if null is returned from WordNet, but it is considered non-vague if gloss is returned instead. A term may be vague due to it contains delimiter characters; thus, a preprocessing step is needed to remove the unnecessary delimiter characters (i.e., underscore, dash, etc.) as they are considered as noise. The preprocessing step is performed by tokenizing a term of a request, av_{Req}, and a term of a policy, av_{Pol}, into fragments of words if they contain tokens separated by delimiter characters. The tokens of av_{Req} are stored into an array, $array_{av_{Req}}$, whereas the tokens of av_{Pol} are stored into an array, $array_{av_{Pol}}$.

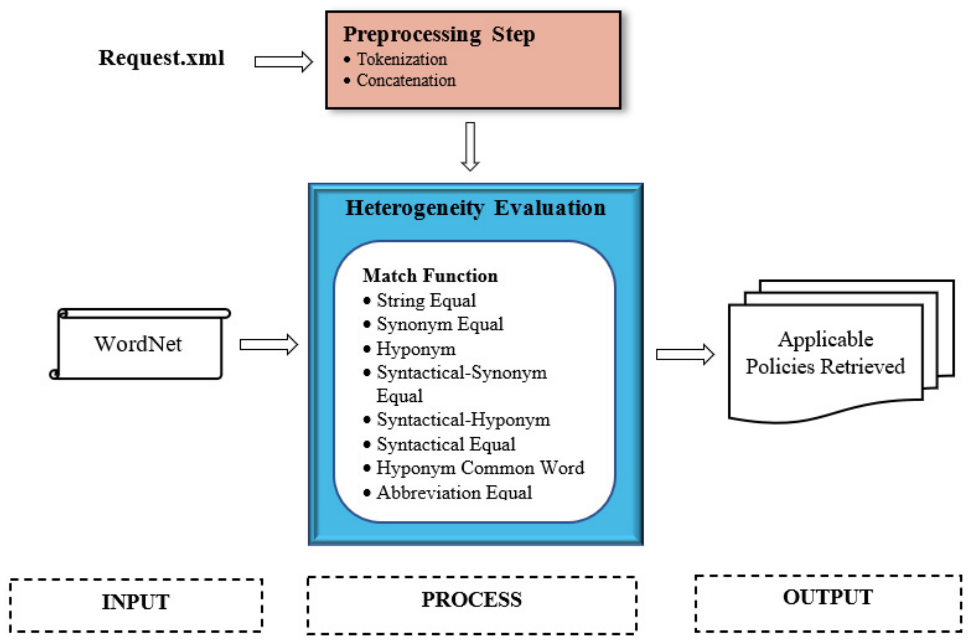

Figure 1. The naming heterogeneity resolution model.

There are two fundamental string concatenation operators in concatenating multiple tokens into a meaningful term. They are space concatenation and abuttal concatenation, as elaborated below:

- Space concatenation is performed on the tokens of the $array_{av_{Req}}$ and $array_{av_{Pol}}$. The tokens of the $array_{av_{Req}}$ are concatenated with an intervening space and stored into $term_{Reqspace}$. $term_{Reqspace}$ is further checked by WordNet as to whether it is a non-vague term. The same process goes for $array_{av_{Pol}}$. For example, the tokens of av_{Req}, *Faculty_Member* and $\{Faculty,\ Member\}$, are concatenated with an intervening space to form a new meaningful term, *Faculty Member*, which is a non-vague term, as gloss is returned from WordNet.
- In contrast, if the new term is a vague term, abuttal concatenation is performed by concatenating the tokens into a single term without an intervening space. Take av_{Pol}, $Undergrad - Class\_rc$ as an example. $Undergrad - Class\_rc$ is tokenized into $\{Undergrad,\ Class,\ rc\}$ and is further concatenated with an intervening space to form a new term, av_{Polnew}, *Undergrad Class rc*. However, *Undergrad Class rc* is apparently a vague term, thus, *Undergrad*, *Class*, and *rc* are concatenated into a single term without an intervening space, *UndergradClassrc*. Intuitively, *N-gram* is applied during the matching process.

Algorithm 1 presents the preprocessing steps before running the matching functions in order to remove the unnecessary delimiters.

The outputs of Algorithm 1 are the new form of both av_{Req} and av_{Pol} denoted by av_{Reqnew} and av_{Polnew}, respectively, which then become the input to the matching functions. A matching function will return a set of tuples of the form $\langle av_{Req}, av_{Pol}, Result\rangle$ where av_{Req} (av_{Pol}) is the term of a request (policy, respectively) in its initial form and *Result* returns *Matched* if av_{Reqnew} matched with av_{Polnew}, while returning *Not Matched* otherwise. We have devised eight matching functions, each cater a different form of a term, namely: *String Equal*, *Synonym Equal*, *Hyponym*, *Syntactical Synonym Equal*, *Syntactical Hyponym*, *Syntactical Equal*, *Hyponym Common Word*, and *Abbreviation Equal*. For *String Equal*, *Synonym Equal*, *Hyponym*, and *Hyponym Common Word*, the similarity score is equal to 1 when the matching functions return a match value and is equal to 0 otherwise. The following sections present how the proposed matching functions work on the string value.

Algorithm 1: Preprocessing Steps Algorithm

```
Input: A term of a $Req$, $av_{Req}$; A term of a $Pol$, $av_{Pol}$
Output: A new term of the $Req$, $av_{Reqnew}$ ; A new term of the $Pol$, $av_{Polnew}$
For $av_{Req}$ in $Req$
If $av_{Req}$ contains delimiters
    Then, tokenize $av_{Req}$ into $s$ fragments and store the $s$ fragments into $array_{av_{Req}}$
    Perform space concatenation on the tokens of $array_{av_{Req}}$ and store it into $term_{Reqspace}$
    If $term_{Reqspace} \in word$ in WordNet
        Then, $av_{Reqnew} = term_{Reqspace}$
    ElseIf $term_{Reqspace} \notin words$ in WordNet
        Perform abuttal concatenation on the tokens of $array_{av_{Req}}$ and store it into $av_{Reqnew}$
ElseIf $av_{Req}$ does not contain delimiters
    Then, $av_{Reqnew} = av_{Req}$
For $av_{Pol}$ in $Pol$
If $av_{Pol}$ contains delimiters
    Then, tokenize $av_{Pol}$ into $t$ fragments and store the $t$ fragments into $array_{av_{Pol}}$
    Perform space concatenation on the tokens of $array_{av_{Pol}}$ and store it into $term_{Polspace}$
    If $term_{Polspace} \in word$ in WordNet
        Then, $av_{Polnew} = term_{Polspace}$
    ElseIf $term_{Polspace} \notin word$ in WordNet
        Perform abuttal concatenation on the tokens of $array_{av_{Pol}}$ and store it into $av_{Polnew}$
ElseIf $av_{Pol}$ does not contain delimiters
    Then, $av_{Polnew} = av_{Pol}$
```

4.1. String Equal

This function aims to find the similarity between two terms by analyzing its length and characters. av_{Reqnew} and av_{Polnew} are considered matched if they are of the same length and all the characters of the two terms are matched exactly. For example, av_{Reqnew}, $Student$ matches exactly with av_{Polnew}, $Student$. Obviously, both of the two terms are string equal match. Algorithm 2 presents the *String Equal* function algorithm.

Algorithm 2: String Equal Function Algorithm

```
Input: A term of a $Req$, $av_{Reqnew}$; A term of a $Pol$, $av_{Polnew}$
Output: $Matched/Not\ Matched$
If ($av_{Reqnew} == av_{Polnew}$)
    Then, $Matched$
Else
    $Not\ Matched$
```

4.2. Synonym Equal

This function attempts to resolve the terminological variation between two non-vague terms by analyzing the synonym relationship, based on WordNet. The proposed function uses WordNet as a dictionary to identify the synonyms of av_{Polnew}. All synonyms of av_{Polnew} are retrieved from WordNet and stored into an array, $array_{Polnewsyn}$. If av_{Reqnew} matches with one of the synonyms in the $array_{Polnewsyn}$, then av_{Reqnew} is matched with

av_{Polnew}. For example, av_{Reqnew}, *Undergraduate* matches exactly with one of the synonyms of av_{Polnew}, *Undergrad*. Algorithm 3 presents the *Synonym Equal* function algorithm.

Algorithm 3: Synonym Equal Function Algorithm

```
Input: A term of a $Req$, $av_{Reqnew}$; A term of a $Pol$, $av_{Polnew}$
Output: $Matched/Not\ Matched$

If ($av_{Reqnew}! = av_{Polnew}$) && ($av_{Reqnew} \in words$ in WordNet) && ( $av_{Polnew} \in words$ in WordNet)
    Retrieve the synonyms of $av_{Polnew}$ from WordNet and store them into $array_{Polnewsyn}$
    If ($av_{Reqnew} \in array_{Polnewsyn}$)
        Then, $Matched$
    Else
        $Not\ Matched$
Else
    $Not\ Matched$
```

4.3. Hyponym

This function aims to resolve the terminological variation between two non-vague terms by analyzing the hyponym relationship, based on WordNet. The proposed function uses WordNet as a dictionary to identify the hyponyms. All hyponyms of av_{Polnew} are retrieved from WordNet and stored into an array, $array_{Polnewhyp}$. If av_{Reqnew} matches with one of the hyponyms in the $array_{Polnewhyp}$, then av_{Reqnew} is matched with av_{Polnew}. For example, av_{Reqnew}, *Undergraduate* matches exactly with one of the hyponyms of av_{Polnew}, *Student*. Algorithm 4 presents the *Hyponym* function algorithm.

Algorithm 4: Hyponym Function Algorithm

```
Input: A term of a $Req$, $av_{Reqnew}$; A term of a $Pol$, $av_{Polnew}$
Output: $Matched/Not\ Matched$

If ($av_{Reqnew}! = av_{Polnew}$) && ($av_{Reqnew} \in words$ in WordNet) && ($av_{Polnew} \in words$ in WordNet)
    Retrieve the hyponyms of $av_{Polnew}$ from WordNet and store them into $array_{Polnewhyp}$
    If ($av_{Reqnew} \in array_{Polnewhyp}$)
        Then, $Matched$
    Else
        Retrieve the hyponyms of $av_{Reqnew}$ from WordNet and store them into $\in array_{Reqnewhyp}$
        If ($av_{Polnew} \in array_{Reqnewhyp}$)
            Then, $Matched$
        Else
            $Not\ Matched$
Else
    $Not\ Matched$
```

4.4. Syntactical Synonym Equal

This function attempts to resolve the syntactic and terminological variations between a vague term and a non-vague term by analyzing the synonym relationship. All synonyms of a non-vague term are retrieved from WordNet and stored into an array, $array_{synter}$. The *N-gram* similarity measure is applied to calculate the similarity score, SC, between the vague term and each synonym of the non-vague term. If SC exceeds the similarity threshold, τ, then both terms are considered matched. The values of SC and τ are between 0 and 1. For example, consider a vague term of av_{Reqnew}, *UndergraduateStudent* and a vague term of av_{Polnew}, $Undergrad$. If *N-gram* with trigram (3) is applied on the strings $UndergraduateStudent$ and $Undergrad$, the SC between both strings is 0.38 and it is only considered matched if τ is set less than or equal to 0.38. Therefore, all synonyms of $Undergrad$ are retrieved and stored into an array, $array_{synter}$, $\{undergraduate\}$. *N-gram* with trigram (3) is applied on the string $UndergraduateStudent$ and each synonym of $Undergrad$. The SC between $UndergraduateStudent$ and one of the synonyms of $Undergrad$, i.e., $undergraduate$, is found to be 0.53 and it is greater than the default value of τ which is 0.5. Thus, av_{Reqnew} is matched with av_{Polnew}. Algorithm 5 presents the *Syntactical Synonym Equal* function algorithm.

4.5. Syntactical Hyponym

This function aims to resolve the syntactic and terminological variations between a vague term and a non-vague term by analyzing the hyponym relationship. All hyponyms of the non-vague term are retrieved from WordNet and stored into an array, $array_{newhypm}$. The *N-gram* similarity measure is applied to calculate the similarity score, SC, between the vague term and each hyponym of the non-vague term. If SC exceeds the similarity threshold, τ, both terms are considered matched. The values of SC and τ are between 0 and 1. For example, consider a vague term of av_{Reqnew}, $AssociateProf$ and a vague term of av_{Polnew}, $Faculty\ Member$. Hyponym relation is transitive [46]. All hyponyms of $Faculty\ Member$ are retrieved and stored into an array, $array_{newhypm}$, $\{professor,\ prof,\ associate\ professor\}$. *N-gram* with trigram (3) is applied on the string $AssociateProf$ and each hyponym of $Faculty\ Member$. The SC between $AssociateProf$ and one of the hyponyms of $Faculty\ Member$, $associate\ professor$, is found to be 0.5 and it is equal to the τ value by default. Thus, av_{Reqnew} is matched with av_{Polnew}. Algorithm 6 presents the *Syntactical Hyponym* function algorithm.

Algorithm 5: Syntactical Synonym Equal Function Algorithm

```
Input:   A term of a $Req$, $av_{Reqnew}$; A term of a $Pol$, $av_{Polnew}$; Similarity threshold, $\tau$
Output:  $Matched/Not\ Matched$

If $(av_{Reqnew} != av_{Polnew})$ && $(av_{Reqnew} \notin words$ in WordNet) && $(av_{Polnew} \in words$ in WordNet)
    Retrieve the synonyms of $av_{Polnew}$ from WordNet and stored them into $array_{synter}$
    For each element of $array_{synter}$
        $SC$ = Apply N-gram between $av_{Reqnew}$ and the element of $array_{synter}$
        If $(SC \geq \tau)$
            Then, $Matched$
            Break
        Else
            $Not\ Matched$
ElseIf $(av_{Reqnew} != av_{Polnew})$ && $(av_{Reqnew} \in words$ in WordNet) && $(av_{Polnew} \notin words$ in WordNet)
    Retrieve the synonyms of $av_{Reqnew}$ from WordNet and stored them into $array_{synter}$
    For each element of $array_{synter}$
        $SC$ = Apply N-gram between $av_{Polnew}$ and the element of$array_{synter}$
        If $(SC \geq \tau)$
            Then, $Matched$
            Break
        Else
            $Not\ Matched$
Else
    $Not\ Matched$
```

Algorithm 6: Syntactical Hyponym Function Algorithm

```
Input:  A term of a $Req$, $av_{Reqnew}$; A term of a $Pol$, $av_{Polnew}$; Similarity threshold, $\tau$
Output:  $Matched/Not\ Matched$

If ($av_{Reqnew}! = av_{Polnew}$) && ($av_{Reqnew} \notin words$ in WordNet) && ($av_{Polnew} \in words$ in WordNet)
    Retrieve the hyponyms of $av_{Polnew}$ from WordNet and stored them into $array_{newhypm}$
    For each element of $array_{newhypm}$
        $SC =$ Apply N-gram between $av_{Reqnew}$ and the element of $array_{newhypm}$
        If ($SC \geq \tau$)
            Then, $Matched$
            Break
        Else
            $Not\ Matched$
ElseIf ($av_{Reqnew}! = av_{Polnew}$) && ($av_{Reqnew} \in words$ in WordNet) && ($av_{Polnew} \notin words$ in WordNet)
    Retrieve the hyponyms of $av_{Reqnew}$ from WordNet and stored them into an array, $array_{newhypm}$
    For each element of $array_{newhypm}$
        $SC =$ Apply N-gram between $av_{Polnew}$ and the element of $array_{newhypm}$
        If ($SC \geq \tau$)
            Then, $Matched$
            Break
        Else
            $Not\ Matched$
Else
    $Not\ Matched$
```

4.6. Syntactical Equal

This function aims to resolve the syntactic variation between two vague terms. The function *N-gram* is applied to calculate the similarity score, *SC*, between av_{Reqnew} and av_{Polnew}. If the *SC* between av_{Reqnew} and av_{Polnew} exceeds the similarity threshold, τ, both terms are considered matched. The values of *SC* and τ are between 0 and 1. For example, consider the vague terms of av_{Reqnew}, *UndergradClass* and av_{Polnew}, *UndergradClassrc*. *N-gram* with trigram (3) is applied on the strings *UndergradClass* and *UndergradClassrc*. The *SC* between both strings is 0.7 and it is greater than τ by default which is 0.5. Thus, *UndergradClass* is matched with *UndergradClassrc*. Algorithm 7 presents the *Syntactical Equal* function algorithm.

Algorithm 7: Syntactical Equal Function Algorithm

```
Input:  A term of a $Req$, $av_{Reqnew}$; A term of a $Pol$, $av_{Polnew}$; Similarity threshold, $\tau$
Output:  $Matched/Not\ Matched$

If ($av_{Reqnew}! = av_{Polnew}$) && ($av_{Reqnew} \notin words$ in WordNet) && ($av_{Polnew} \notin words$ in WordNet)
    $SC =$ Apply N-gram between $av_{Reqnew}$ and $av_{Polnew}$
    If ($SC \geq \tau$)
        Then, $Matched$
    Else
        $Not\ Matched$
Else
    $Not\ Matched$
```

4.7. Hyponym Common Word

This function attempts to resolve the terminological variation between two terms by analyzing the hyponym relationship. This function checks whether a word appears at the beginning or at the end of another word, which is in the form of a compound noun. The common word between av_{Reqnew} and av_{Polnew} could imply the semantic measure between them. If there is a common substring, its position will provide the evidence for the existence of a hyponymy [44]. Therefore, the same concept is applied to this function by assuming there is a hyponym relationship between the two terms if both terms share a common word. If av_{Polnew} is a common word of av_{Reqnew}, then av_{Polnew} is more general than av_{Reqnew}. In other words, av_{Reqnew} is more specific than av_{Polnew}. The length of av_{Reqnew} and av_{Polnew} should be greater than three before the function is performed to avoid invalid hits returned by this function (e.g., *TeachingCourse* is not relevant for the term *Tea*) [47]. If the length of a term is shorter than the other term, it is called short term, otherwise it is called long term. Each token of the long term is stored into an array, $array_{longterm}$. If the short term matches with one of the elements in the $array_{longterm}$, then av_{Reqnew} is matched with av_{Polnew}. For example, consider av_{Reqnew}, $Graduate\ Student$ as a long term and av_{Polnew}, $Student$ as a short term. Each token of $Graduate\ Student$ is stored into the $array_{longterm}$, $\{Graduate,\ Student\}$. The av_{Polnew}, $Student$ is found to match with one of the elements of the $array_{longterm}$, $Student$. Thus, av_{Reqnew} is matched with av_{Polnew}. Algorithm 8 presents the *Hyponym Common Word* function algorithm.

Algorithm 8: Hyponym Common Word Function Algorithm

```
Input: A term of a $Req$, $av_{Reqnew}$; A term of a $Pol$, $av_{Polnew}$
Output: $Matched/Not\ Matched$

If ($av_{Reqnew} != av_{Polnew}$) && ($av_{Reqnew}.length() > 3$) && ($av_{Polnew}.length() > 3$)
    If $av_{Reqnew}.length \geq av_{Polnew}.length$
    $short\ term = av_{Polnew}$,
    $long\ term = av_{Reqnew}$
    Else
        $short\ term = av_{Reqnew}$,
        $long\ term = av_{Polnew}$
Tokenize the $long\ term$ into $u$ fragments and store the $u$ fragments into an array,
$array_{longterm}$
    If ($short\ term \in array_{longterm}$)
      Then, $Matched$
    Else
        $Not\ Matched$
 Else
     $Not\ Matched$
```

4.8. Abbreviation Equal

This function attempts to resolve the syntactic variation between two terms which may arise due to the short forms. If the length of a term is shorter than the other term, it is called short form, otherwise it is called long form. Every character in the short form must match a character in the long form, and the matched characters in the long form must be in the same order as the characters in the short form. An extraction process is performed on the first letter of each token in the long form and further concatenated into a new term, $term_{EValue}$. *N-gram* is then applied to calculate the similarity score, SC, between the term, $term_{EValue}$, and the short form. If SC exceeds the similarity threshold, τ, then both terms are considered matched. The values of SC and τ are between 0 and 1. For example, consider av_{Reqnew}, *TeachingAssistant* as a long form and av_{Polnew}, *TA* as a short form. Each token of $TeachingAssistant$ is stored into an array, $array_{longterm}$, $\{Teaching, Assistant\}$. Then, the initial letter of each token of $TeachingAssistant$ is extracted and concatenated into a single new term, $term_{EValue}$, TA. The SC of $term_{EValue}$, TA and the short form, TA is 1.0 and it is greater than τ by default, which is 0.5. Thus, $TeachingAssistant$ is matched with TA. Algorithm 9 presents the *Abbreviation Equal* function algorithm.

Algorithm 9: Abbreviation Equal Function Algorithm

```
Input: A term of a $Req$, $av_{Reqnew}$; A term of a $Pol$, $av_{Polnew}$; Similarity threshold, $\tau$
Output: $Matched/Not\ Matched$

If $av_{Reqnew}.length \geq av_{Polnew}.length$
    $short\ term = av_{Polnew}$,
    $long\ term = av_{Reqnew}$
Else
    $short\ term = av_{Reqnew}$,
    $long\ term = av_{Polnew}$
Tokenize the word of $long\ term$ into $u$ fragments and store the $u$ fragments into an array,
$array_{longterm}$
Extract the initial letter of each token of $array_{longterm}$ and concatenate them into a new term,
$term_{EValue}$
$SC$ =Apply $N$-$gram$ between $term_{EValue}$ and $short\ term$
If $(SC \geq \tau)$
    Then, $Matched$
Else
    $Not\ Matched$
```

5. Results and Discussion

Several experiments were conducted to measure the accuracy of the proposed matching functions in resolving naming heterogeneity that occurs between the attribute values of a request and a policy. These experiments aimed to show the strengths and weaknesses of the proposed solution in resolving naming heterogeneity between a request and a policy with respect to the subject, resource, action, and condition.

The proposed matching functions were implemented in Java and XACML policy language with Java 1.6.0 10. The match task was first conducted manually by three professional human experts who were either familiar with database management or English linguistics. Sun's XACML implementation [11] was chosen as the comparison since several related works, such as [8–10,48], have selected Sun's XACML implementation for their results comparison, with the strong justification that Sun's XACML implementation is an open source. However, these works focused on the efficiency of their engine in processing the requests, whereas this work focuses on the accuracy of resolving naming heterogeneity between a request and a policy.

In this work, six sets of XACML policies (http://sourceforge.net/projects/xacmlpdp/ (accessed on 29 March 2017)) were taken from [9] that have been designed for a university and a conference management domain, namely: *CodeA*, *CodeB*, *CodeC*, *CodeD*, *Continue-a*, and *Continue-b*. *Continue-a* and *Continue-b* are designed for a conference management while *CodeA*, *CodeB*, *CodeC*, and *CodeD* are designed for a real-world web application supporting the university domain. Another two sets of policies that have been analyzed were taken from [49]. Those policies are based on the RBAC model and are designed for a university (http://www3.cs.stonybrook.edu/stoller/ccs2007/university-policy.txt (accessed on 29 March 2017)) and a health care (http://www3.cs.stonybrook.edu/stoller/ccs2007/healthcare.txt (accessed on 29 March 2017)) institution. The RBAC policies are presented in the syntax and structure of XACML with positive effects since negative authorizations are not supported in the RBAC model [3]. These sets of policies were modified by adding additional condition context since initially these policies do not contain condition context. This modification was necessary for the purpose of evaluating the accuracy of the proposed matching functions. A request-generation technique [50] was used to generate 10,000 requests at random since most of the real-world systems use less than 10,000 policies [16]. Eight sets of the modified XACML policy datasets mentioned above were used as the source to generate the random requests. Since there is a distinct lack of real request datasets in a distributed environment, the domain ontologies related to university (http://swat.cse.lehigh.edu/onto/univ-bench.owl (accessed on 29 March 2017)), conference management (http://data.semanticweb.org/ns/swc/swc2009-05-09.html (accessed on 29 March 2017)), and health care institution (https://loinc.org/discussion-documents/document-ontology/loinc-document-ontology-axisvalues?force_toc:int=1 (accessed on 29 March 2017)) domains were selected as the source to generate the random requests.

In order to measure the accuracy of the matching results in each experiment, Precision (*P*), Recall (*R*), and F-Measure (*F*), originating from the information retrieval field, were used [43]. Each experiment was conducted five times. The match results of *P*, *R*, and *F* in matching the attribute values of a request and a policy by the proposed solution and the Sun's XACML implementation were compared to the real match results obtained by the human experts. The results were analyzed at various values of similarity thresholds. The match results of *P*, *R*, and *F* are presented in Tables 11–18.

Table 11. Precision (*P*), Recall (*R*), and F-measure (*F*) of the proposed solution with different similarity thresholds and the Sun's XACML implementation for the *CodeA* policy.

Evaluation Metric	**Attributes**	**Percentage (%)**						**Improvement [Lowest, Highest]**
		Proposed Solution					**Sun's XACML Implementation**	
		Similarity Threshold (τ)						
		0.2	**0.4**	**0.6**	**0.8**	**1.0**		
Precision (*P*)	Subject	100	100	100	100	100	100	[0, 0]
	Resource	56.57 *	56.57 *	56.57 *	100	100	100	[−43.43, 0]
	Action	100	100	100	100	100	100	[0, 0]
	Condition	86.67 *	93.55 *	100	100	100	100	[−13.33, 0]
Recall (*R*)	Subject	52.22	42.22	42.22	42.22	42.22	5.56	[36.66, 46.66]
	Resource	100	100	100	100	100	100	[0, 0]
	Action	88.33	56.67	43.33	43.33	43.33	20.00	[23.33, 68.33]
	Condition	33.12	19.05	18.47	18.47	18.47	1.27	[17.20, 31.85]
F-Measure (*F*)	Subject	68.61	59.38	59.38	59.38	59.38	10.53	[48.85, 58.08]
	Resource	72.26 *	72.26 *	72.26 *	100	100	100	[−27.74, 0]
	Action	93.81	72.34	60.47	60.47	60.47	33.33	[27.14, 60.48]
	Condition	47.93	31.65	31.18	31.18	31.18	2.52	[28.66, 45.41]

* The proposed solution match results are worse than the Sun's XACML implementation match results.

Table 12. Precision (*P*), Recall (*R*), and F-measure (*F*) of the proposed solution with different similarity thresholds and the Sun's XACML implementation for the *CodeB* policy.

Evaluation Metric	**Attributes**	**Percentage (%)**						**Improvement [Lowest, Highest]**
		Proposed Solution					**Sun's XACML Implementation**	
		Similarity Threshold (τ)						
		0.2	**0.4**	**0.6**	**0.8**	**1.0**		
Precision (*P*)	Subject	100	100	100	100	100	100	[0, 0]
	Resource	53.94 *	53.94 *	53.94 *	100	100	100	[−46.06, 0]
	Action	100	100	100	100	100	100	[0, 0]
	Condition	97.53 *	100	100	100	100	100	[−2.47, 0]

Table 12. *Cont.*

Evaluation Metric	Attributes	Percentage (%)						
		Proposed Solution					Sun's XACML Implementation	Improvement [Lowest, Highest]
		Similarity Threshold (τ)						
		0.2	0.4	0.6	0.8	1.0		
Recall (*R*)	Subject	53.92	45.10	45.10	45.10	45.10	12.70	[32.40, 41.22]
	Resource	100	100	100	100	100	100	[0, 0]
	Action	93.20	56.31	40.78	40.78	40.78	13.59	[27.19, 79.61]
	Condition	31.98	25.51	25.51	25.51	25.51	5.26	[20.25, 26.72]
F-Measure (*F*)	Subject	70.06	62.16	62.16	62.16	62.16	22.61	[39.55, 47.45]
	Resource	70.08 *	70.08 *	70.08 *	100	100	100	[−29.92, 0]
	Action	96.48	72.05	57.93	57.93	57.93	23.93	[34.00, 72.55]
	Condition	48.17	40.65	40.65	40.65	40.65	10.00	[30.65, 38.17]

* The proposed solution match results are worse than the Sun's XACML implementation match result.

Table 13. Precision (*P*), Recall (*R*), and F-measure (*F*) of the proposed solution with different similarity thresholds and the Sun's XACML implementation for the *CodeC* policy.

Evaluation Metric	Attributes	Percentage (%)						
		Proposed Solution					Sun's XACML Implementation	Improvement [Lowest, Highest]
		Similarity Threshold (τ)						
		0.2	0.4	0.6	0.8	1.0		
Precision (*P*)	Subject	100	100	100	100	100	100	[0, 0]
	Resource	50.00 *	50.00 *	50.00	100	100	100	[−50.00, 0]
	Action	100	100	100	100	100	100	[0, 0]
	Condition	92.36 *	98.04 *	100	100	100	100	[−7.64, 0]
Recall (*R*)	Subject	53.92	45.10	45.10	45.10	45.10	12.75	[32.35, 41.17]
	Resource	100	100	100	100	100	100	[0, 0]
	Action	95.21	56.16	39.73	39.73	39.73	10.96	[28.77, 84.25]
	Condition	60.92	46.01	42.02	42.02	42.02	11.34	[30.68, 49.58]
F-Measure (*F*)	Subject	70.06	62.16	62.16	62.16	62.16	22.61	[39.55, 47.45]
	Resource	66.67 *	66.67 *	66.67 *	100	100	100	[−33.33, 0]
	Action	97.54	71.93	56.86	56.86	56.86	19.75	[37.11, 77.79]
	Condition	73.42	62.63	59.17	59.17	59.17	20.38	[38.79, 53.04]

* The proposed solution match results are worse than the Sun's XACML implementation match results.

Table 14. Precision (*P*), Recall (*R*), and F-measure (*F*) of the proposed solution with different similarity thresholds and the Sun's XACML implementation for the *CodeD* policy.

Evaluation Metric	Attributes	Percentage (%)						
		Proposed Solution					Sun's XACML Implementation	Improvement [Lowest, Highest]
		Similarity Threshold (τ)						
		0.2	0.4	0.6	0.8	1.0		
Precision (*P*)	Subject	100	100	100	100	100	100	[0, 0]
	Resource	53.94 *	53.94 *	53.94 *	100	100	100	[−46.06, 0]
	Action	100	100	100	100	100	100	[0, 0]
	Condition	93.17 *	98.36 *	100	100	100	100	[−6.83, 0]
Recall (*R*)	Subject	38.51	31.76	31.76	31.76	31.76	9.46	[22.30, 29.05]
	Resource	100	100	100	100	100	100	[0, 0]
	Action	91.41	56.44	41.72	41.72	41.72	15.95	[25.77, 75.46]
	Condition	70.22	47.08	44.12	44.12	44.12	13.60	[30.52, 56.62]

Table 14. *Cont.*

Evaluation Metric	Attributes	Percentage (%)						Improvement [Lowest, Highest]
		Proposed Solution					Sun's XACML Implementation	
		Similarity Threshold (τ)						
		0.2	0.4	0.6	0.8	1.0		
F-Measure (*F*)	Subject	55.61	48.21	48.21	48.21	48.21	17.28	[30.93, 38.33]
	Resource	70.08 *	70.08 *	70.08 *	100	100	100	[−29.92, 0]
	Action	95.51	72.16	58.87	58.87	58.87	27.51	[31.36, 68.00]
	Condition	80.08	63.68	61.22	61.22	61.22	23.95	[37.27, 56.13]

* The proposed solution match results are worse than the Sun's XACML implementation match results.

Table 15. Precision (*P*), Recall (*R*), and F-measure (*F*) of the proposed solution with different similarity thresholds and the Sun's XACML implementation for the UniversityStoller policy.

Evaluation Metric	Attributes	Percentage (%)						Improvement [Lowest, Highest]
		Proposed Solution					Sun's XACML Implementation	
		Similarity Threshold (τ)						
		0.2	0.4	0.6	0.8	1.0		
Precision (*P*)	Subject	100	100	100	100	100	100	[0, 0]
	Resource	65.70 *	91.67 *	100	100	100	100	[−34.30, 0]
	Action	61.07 *	100	100	100	100	100	[−38.93, 0]
	Condition	89.69 *	92.85 *	98.98 *	100	100	100	[−10.31, 0]
Recall (*R*)	Subject	50.52	35.01	34.98	34.98	34.98	17.01	[17.97, 33.51]
	Resource	43.86	36.11	33.25	33.25	33.25	31.33	[1.92, 12.53]
	Action	61.66	59.45	59.45	59.45	59.45	40.55	[18.90, 21.11]
	Condition	47.98	39.04	38.02	37.04	37.04	9.13	[27.91, 38.85]
F-Measure (*F*)	Subject	67.12	51.86	51.83	51.83	51.83	29.07	[22.76, 38.05]
	Resource	52.60	51.81	49.91	49.91	49.91	47.71	[2.20, 4.89]
	Action	61.36	74.57	74.57	74.57	74.57	57.70	[3.66, 16.87]
	Condition	62.52	54.97	54.93	54.06	54.06	16.73	[37.33, 45.79]

* The proposed solution match results are worse than the Sun's XACML implementation match results.

Table 16. Precision (*P*), Recall (*R*), and F-measure (*F*) of the proposed solution with different similarity thresholds and the Sun's XACML implementation for the *Continue-a* policy.

Evaluation Metric	Attributes	Percentage (%)						Improvement [Lowest, Highest]
		Proposed Solution					Sun's XACML Implementation	
		Similarity Threshold (τ)						
		0.2	0.4	0.6	0.8	1.0		
Precision (*P*)	Subject	100	100	100	100	100	100	[0, 0]
	Resource	13.33 *	62.07 *	82.35 *	100	100	100	[−86.67, 0]
	Action	100	100	100	100	100	100	[0, 0]
	Condition	100	100	100	100	100	100	[0, 0]
Recall (*R*)	Subject	66.12	66.12	66.12	66.12	66.12	50.00	[16.12, 16.12]
	Resource	100	100	77.78	77.78	77.78	47.22	[30.56, 52.78]
	Action	74.07	74.07	74.07	74.07	74.07	74.07	[0, 0]
	Condition	100	100	100	100	100	100	[0, 0]
F-Measure (*F*)	Subject	79.60	79.60	79.60	79.60	79.60	99.00	[19.40, 19.40]
	Resource	23.53	76.60	80.00	97.15	97.15	64.15	[−40.62, 33.00]
	Action	85.10	85.10	85.10	85.10	85.10	85.10	[0, 0]
	Condition	100	100	100	100	100	100	[0, 0]

* The proposed solution match results are worse than the Sun's XACML implementation match results.

Table 17. Precision (*P*), Recall (*R*), and F-measure (*F*) of the proposed solution with different similarity thresholds and the Sun's XACML implementation for the *Continue-b* policy.

Evaluation Metric	**Attributes**	**Percentage (%)**						**Improvement [Lowest, Highest]**
		Proposed Solution					**Sun's XACML Implementation**	
		Similarity Threshold (τ)						
		0.2	**0.4**	**0.6**	**0.8**	**1.0**		
Precision (*P*)	Subject	100	100	100	100	100	100	[0, 0]
	Resource	13.33 *	62.07 *	82.35 *	100	100	100	[−86.67, 0]
	Action	100	100	100	100	100	100	[0, 0]
	Condition	100	100	100	100	100	100	[0, 0]
Recall (*R*)	Subject	66.12	66.12	66.12	66.12	66.12	50.00	[16.12, 16.12]
	Resource	100	100	77.78	77.78	77.78	47.22	[30.56, 52.78]
	Action	74.07	74.07	74.07	74.07	74.07	74.07	[0, 0]
	Condition	100	100	100	100	100	100	[0, 0]
F-Measure (*F*)	Subject	79.60	79.60	79.60	79.60	79.60	99.00	[19.40, 19.40]
	Resource	23.53	76.60	80.00	97.15	97.15	64.15	[−40.62, 33.00]
	Action	85.10	85.10	85.10	85.10	85.10	85.10	[0, 0]
	Condition	100	100	100	100	100	100	[0, 0]

* The proposed solution match results are worse than the Sun's XACML implementation match results.

Table 18. Precision (*P*), Recall (*R*), and F-measure (*F*) of the proposed solution with different similarity thresholds and the Sun's XACML implementation for the HealthCare policy.

Evaluation Metric	**Attributes**	**Percentage (%)**						**Improvement [Lowest, Highest]**
		Proposed Solution					**Sun's XACML Implementation**	
		Similarity Threshold (τ)						
		0.2	**0.4**	**0.6**	**0.8**	**1.0**		
Precision (*P*)	Subject	100	100	100	100	100	100	[0, 0]
	Resource	68.20 *	69.53 *	100	100	100	100	[−31.80, 0]
	Action	60.71 *	100	100	100	100	100	[−39.29, 0]
	Condition	81.04 *	97.35 *	100	100	100	100	[−18.96, 0]
Recall (*R*)	Subject	88.37	86.82	82.95	82.95	82.95	45.51	[37.44, 42.86]
	Resource	26.56	17.25	15.45	15.45	15.45	13.76	[1.69, 12.80]
	Action	80.68	80.68	80.68	80.68	80.68	80.68	[0, 0]
	Condition	45.13	42.69	42.69	42.69	42.69	23.20	[19.49, 21.93]
F-Measure (*F*)	Subject	93.83	92.95	90.68	90.68	90.68	63.49	[27.19, 30.34]
	Resource	38.44	27.53	26.76	26.76	26.76	24.19	[2.57, 14.25]
	Action	69.28 *	89.31	89.31	89.31	89.31	89.31	[−20.03, 0]
	Condition	57.97	59.35	59.84	59.84	59.84	37.66	[20.31, 22.18]

* The proposed solution match results are worse than the Sun's XACML implementation match results.

The *Improvement* column in these tables presents the range of improvement achieved by the proposed solution as compared with the Sun's XACML implementation with regard to *P*, *R*, and *F*. The range of improvement is presented by [*Lowest*, *Highest*], where *Lowest* = the lowest value of *P* (*R* or *F*) achieved by the proposed solution minus the value of *P* (*R* or *F*) achieved by the Sun's XACML implementation and *Highest* = the highest value of *P* (*R* or *F*) achieved by the proposed solution minus the value of *P* (*R* or *F*) achieved by the Sun's XACML implementation.

As shown in these tables, the Sun's XACML implementation is able to attain perfect *P* in terms of matching the attribute values of subject, resource, action, and condition of a request and a policy. This indicates that the Sun's XACML implementation never retrieved a false positive, thus, zero false positive was produced, and all the match results returned by the Sun's XACML implementation were true positives, which are the same results as those produced by the human experts. Nevertheless, the Sun's XACML implementation, which adopted the simple string equal matching function, will only return 0 if the strings are different and 1 if they are exactly the same; thus, it does not take into consideration the naming heterogeneity issues. Therefore, the number of false negatives of the Sun's XACML implementation is higher than the proposed solution. As a consequence, the R achieved by the Sun's XACML implementation is lower compared to the R achieved by the proposed solution because most of the match results between the attribute values of subject, resource, action, and condition of a request and a policy were not returned by the Sun's XACML implementation.

It is observed that in the proposed solution, the result of P increased, and the result of R decreased, when the similarity threshold is set to a higher value. The proposed solution returned the same results of P, R, and F for different similarity thresholds for *Continue-a* and *Continue-b* policies, since both policies used the same attribute values. The only difference between *Continue-a* and *Continue-b* is the number of policies. It was also observed that there are two cases which have caused the proposed solution unable to achieve perfect precision in matching the attribute values of a request and a policy.

- The proposed solution could not produce accurate match results when there are similarities in terms of characters presented in the terms being matched while these terms are actually not matched. For example, the terms $ExternalGrades$ and $InternalGrades$ that appeared in the *Continue-a* policy are considered matched in the proposed solution. The application of *N-gram* with trigram (3) on the strings $ExternalGrades$ and $InternalGrades$ gained a similarity score of 0.6, which satisfied the similarity threshold when it was set to at least 0.6. However, $ExternalGrades$ and $InternalGrades$ are considered not matched by the human experts. Referring to Tables 16 and 17, the proposed solution achieved a low value of P in matching the resource attribute of a request and a policy when the similarity threshold is 0.2. This is because the terms in the resource attribute of *Continue-a* and *Continue-b* policies have some similarities in terms of characters presented, thus making *N-gram* produce a higher similarity score than the similarity threshold 0.2 but lower than 0.4. In another example, $Paper - review - content\_rc$ and $Paper - review - info - submissionStatus\_rc$ are considered matched by the proposed solution since the similarity score of these terms is 0.27, which is higher than the similarity threshold 0.2. However, $Paper - review - content\_rc$ and $Paper - review - info - submissionStatus\_rc$ are considered not matched by the human experts. Thus, the proposed solution produced a false positive in this case.
- The proposed solution failed to match the terms that contain semantic relationship but do not have similarities in terms of characters presented. However, these terms are in fact matched. For example, $Research\ Assistant$ in a request and $Faculty\ Member$ in a policy. In this case, the proposed solution returned false match. Based on the human experts, $Research\ Assistant$ is a hyponym of $Faculty\ Member$.

From the experiments, it is obvious that the higher the similarity threshold, the higher the percentage of P in matching the attribute values of a request and a policy. Furthermore, the proposed solution achieved higher percentages of R and F compared with the Sun's XACML implementation. This is due to the fact that *N-gram* and WordNet are utilized in the proposed matching functions.

For most of the datasets, the proposed solution obtained negative improvement based on *Lowest* calculation with respect to P in matching the resource, action, and condition attribute of a request and a policy. This is because the Sun's XACML implementation supports simple string equal matching function and thus produced zero false positive, while the number of false positives produced by the proposed solution exceeds the number of true positives. The proposed solution achieved the highest negative improvement in *Continue-a* and *Continue-b* datasets. This is because among the datasets, *Continue-a* and *Continue-b* datasets contained the largest number of attribute values which have similarities in terms of characters but are actually not matched; thereby, the number of false positives that is produced by the proposed solution is the highest in the *Continue-a* and *Continue-b* datasets. Considering naming heterogeneity, the proposed solution resulted in lower improvement value in *P* but higher improvement value in *R* and *F* compared with the Sun's XACML implementation. The reason is that the Sun's XACML implementation is restrictive to simple string equal matching, which does not consider functions that can resolve naming heterogeneity in matching the attribute value of a request and a policy.

In addition, the proposed solution resulted in no improvement based on the *Highest* calculation with respect to *P* for all sets of policies. This is because the proposed solution achieved 100% of *P* in matching the attribute values of a request and a policy when the similarity threshold is set to a higher value. The Sun's XACML implementation also achieved 100% of *P* for all sets of policies since the Sun's XACML implementation never gained false positives. However, since the proposed solution is able to resolve naming heterogeneity, the number of false negatives of the proposed solution is lower than the Sun's XACML implementation. Thus, the *R* and *F* obtained by the proposed solution for most of the policies are higher than the *R* and *F* obtained by the Sun's XACML implementation. Therefore, we can conclude that the proposed solution is better compared with the Sun's XACML implementation.

Figures 2 and 3 present the improvement based on the *Lowest* calculation and *Highest* calculation, respectively, in retrieving the application policies achieved by the proposed solution, as compared with the Sun's XACML implementation. Based on Figure 2, we observed that there was no improvement made in terms of *P* by the proposed solution and, for most of the datasets, the proposed solution obtained negative improvement. The Sun's XACML implementation never retrieved false positives in retrieving the applicable policies since the Sun's XACML implementation supported simple string equal matching function that does not consider naming heterogeneity; all the matches returned by the Sun's XACML implementation in matching the attribute values of the requests and the policies are true positives, which are the same matched results as those produced by the human experts.

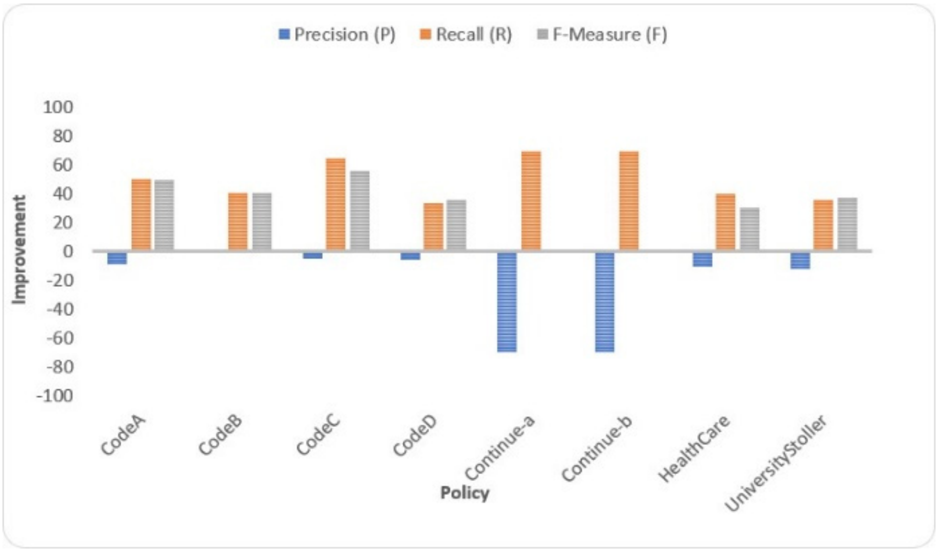

Figure 2. The improvement based on the lowest calculation in retrieving the applicable policies achieved by the proposed solution as compared with the Sun's XACML implementation.

However, the number of false negatives of the Sun's XACML implementation in retrieving the applicable policies is higher than the proposed solution. The reason is that the proposed solution is able to resolve the naming heterogeneity; thus, the proposed solution could reduce the number of false negatives in retrieving the applicable policies. This caused the *R* and *F* achieved by the proposed solution higher than the *R* and *F* achieved by the Sun's XACML implementation. Thus, the proposed solution still outperforms the Sun's XACML implementation in terms of *R* and *F* for all sets of policies.

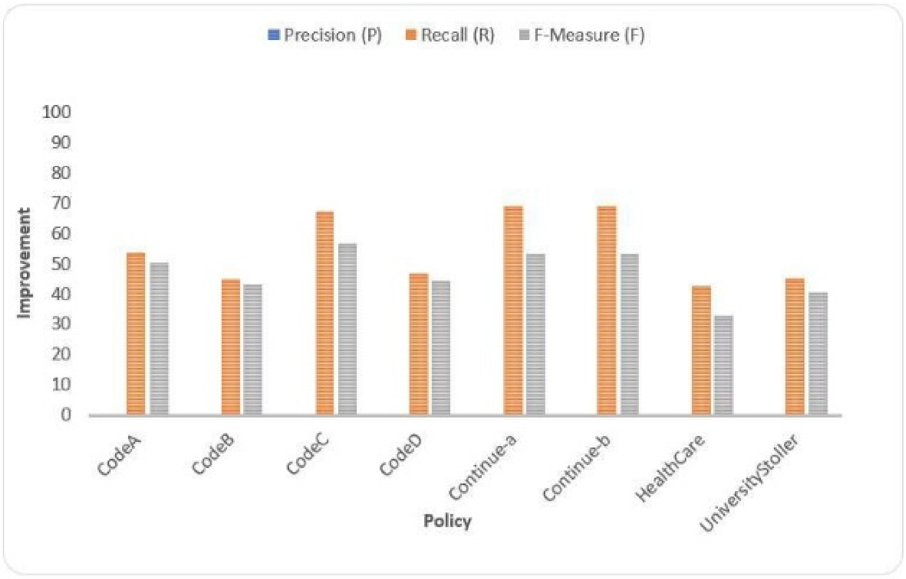

Figure 3. The improvement based on the highest calculation in retrieving the applicable policies achieved by the proposed solution as compared with the Sun's XACML implementation.

Based on Figure 3, we observed that the proposed solution obtained no improvement in terms of *P* for all sets of policies. This is because the proposed solution with similarity threshold set to a higher value and the Sun's XACML implementation, both achieved 100% of *P* in retrieving the applicable policies. However, since the proposed solution is able to resolve the naming heterogeneity, the number of false negatives of the proposed solution in retrieving the applicable policies is lower than the Sun's XACML implementation. This made the *R* and *F* obtained by the proposed solution higher than the *R* and *F* obtained by the Sun's XACML implementation. Therefore, the proposed solution still performs better than the Sun's XACML implementation in terms of *R* and *F* for all sets of policies.

6. Conclusions

This research addresses the significant need in resolving naming heterogeneity for XACML policy evaluation. The proposed matching functions are proven to be capable of resolving naming heterogeneity between the attribute values of a request and a policy during policy evaluation by considering both the syntactic and terminological variations. The proposed solution achieved higher percentage of *P* and *F* when the similarity threshold is set to a higher value. Besides, it also achieved higher percentage of *R* compared with the Sun's XACML implementation. Various experiments have been accomplished, and the results confirmed that our proposed solution has significantly outperformed the Sun's XACML implementation for the six sets of XACML policies that have been considered in the work. Most importantly, the results show that the improvement made by our solution in terms of Recall and F-measure, compared with the Sun's XACML implementation, reached up to 70% and 57%, respectively.

The proposed solution can be further enhanced by considering other factors which could affect the authorization decisions, such as obligations, in which some actions should be launched once certain conditions are satisfied. Besides, the issue of providing a secure mapping of XACML policies and rules for relations with encrypted data using symmetric keys needs to be investigated. A design XML encryption algorithm, which uses symmetrical or asymmetrical keys to achieve the element level encryption before identifying policies and user rules for the encrypted XML elements, is required. Last but not least, further enhancement to the proposed solution in this area can be carried out by investigating the spatial context of a request and a policy which is organized based on the logical data model to be used by the geographic information system (GIS).

Author Contributions: Conceptualization, T.P.K., H.I., F.S. and N.I.U.; methodology, T.P.K. and H.I.; validation, T.P.K., H.I. and A.A.A.; formal analysis, T.P.K., H.I., F.S. and N.I.U.; investigation, T.P.K., H.I., F.S., N.I.U. and A.A.A.; resources, T.P.K. and H.I.; writing—original draft preparation, T.P.K. and H.I.; writing—review and editing, H.I., F.S., N.I.U. and A.A.A.; visualization, T.P.K. and H.I.; supervision, H.I., F.S., N.I.U. and A.A.A.; project administration, H.I.; funding acquisition, H.I. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Universiti Putra Malaysia under the Journal Publishing Initiative Year 2020 scheme (9053006).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank Universiti Putra Malaysia for funding this research work. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies. We thank the anonymous reviewers for their comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tejada, S.; Knoblock, C.A.; Minton, S. Learning object identification rules for information integration. *Inf. Syst.* **2001**, *26*, 607–633. [CrossRef]
2. Thilakanathan, D.; Chen, S.; Nepal, S.; Calvo, R. SafeProtect: Controlled Data Sharing With User-Defined Policies in Cloud-Based Collaborative Environment. *IEEE Trans. Emerg. Top. Comput.* **2015**, *4*, 301–315. [CrossRef]
3. Toosi, A.N.; Calheiros, R.N.; Buyya, R. Interconnected Cloud Computing Environments: Challenges, Taxonomy, and Survey. *J. ACM Comput. Surv.* **2014**, *7*, 1–47. [CrossRef]
4. Trivellato, D.; Zannone, N.; Glaundrup, M.; Skowronek, J.; Etalle, S. A Semantic Security Framework for Systems of Systems. *Int. J. Coop. Inf. Syst.* **2013**, *22*, 1–35. [CrossRef]
5. Trivellato, D.; Spiessens, F.; Zannone, N.; Etalle, S. POLIPO: Policies & Ontologies for Interoperability, Portability, and Autonomy. In Proceedings of the 10th IEEE International Conference on Policies for Distributed Systems and Networks (POLICY), London, UK, 20–22 July 2009; pp. 110–113.
6. Castano, S.; Ferrara, A.; Montanelli, S.; Racca, G. Semantic Information Interoperability in Open Networked Systems. In Proceedings of the International Conference on Semantics of a Networked World (LCSNW), Paris, France, 17–19 June 2004; pp. 215–230.
7. Drozdowicz, M.; Ganzha, M.; Paprzycki, M. Semantically Enriched Data Access Policies in eHealth. *J. Med. Syst.* **2016**, *40*, 238. [CrossRef]
8. Ammar, N.; Malik, Z.; Bertino, E.; Rezgui, A. XACML Policy Evaluation with Dynamic Context Handling. *J. IEEE Trans. Knowl. Data Eng.* **2015**, *27*, 2575–2588. [CrossRef]
9. Liu, A.X.; Chen, F.; Hwang, J.; Xie, T. Designing Fast and Scalable XACML Policy Evaluation Engines. *IEEE Trans. Comput.* **2010**, *60*, 1802–1817. [CrossRef]
10. Ngo, C.; Demchenko, Y.; de Laat, C. Decision Diagrams for XACML Policy Evaluation and Management. *J. Comput. Secur.* **2015**, *49*, 1–16. [CrossRef]
11. Proctor, S. Sun's XACML Implementation. 2004. Available online: http://sunxacml.sourceforge.net (accessed on 29 March 2017).
12. Ciuciu, I.; Zhao, G.; Chadwick, D.W.; Reul, Q.; Meersman, R.; Vasquez, C.; Hibbert, M.; Winfield, S.; Kirkham, T. Ontology based Interoperation for Securely Shared Services: Security Concept Matching for Authorization Policy Interoperability. In Proceedings of the 4th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Paris, France, 7–10 February 2011; pp. 1–5.
13. Ferrini, R.; Bertino, E. Supporting RBAC with XACML+OWL. In Proceedings of the 14th ACM Symposium on Access Control Models and Technologies (SACMAT), Stresa, Italy, 3–5 June 2009; pp. 145–154.
14. Hu, L.; Ying, S.; Jia, X.; Zhao, K. Towards an Approach of Semantic Access Control for Cloud Computing. In Proceedings of the 1st IEEE International Conference on Cloud Computing, Beijing, China, 1–4 December 2009; pp. 145–156.
15. Husain, M.F.; Al-Khateeb, T.; Alam, M.; Khan, L. Ontology based policy interoperability in geo-spatial domain. *Comput. Stand. Interfaces* **2011**, *33*, 214–219. [CrossRef]

16. Mohan, A.; Blough, D.M.; Kurc, T.; Post, A.; Saltz, J. Detection of Conflicts and Inconsistencies in Taxonomy based Authorization Policies. In Proceedings of the 2011 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Atlanta, GA, USA, 12–15 November 2011; pp. 590–594.
17. Priebe, T.; Dobmeier, W.; Schläger, C.; Kamprath, N. Supporting Attribute based Access Control in Authorization and Authentication Infrastructures with Ontologies. *J. Softw.* **2007**, *2*, 27–38. [CrossRef]
18. Takabi, H. A Semantic based Policy Management Framework for Cloud Computing Environments. Ph.D. Thesis, University of Pittsburgh, Pittsburgh, PA, USA, 12 July 2013.
19. Zhao, H. Security Policy Definition and Enforcement in Distributed Systems. Ph.D. Thesis, Columbia University, New York, NY, USA, 12 September 2012.
20. Dia, O.A.; Farkas, C. A Practical Framework for Policy Composition and Conflict Resolution. *Int. J. Secur. Softw. Eng.* **2012**, *3*, 1–26. [CrossRef]
21. Duan, L.; Zhang, Y.; Chen, S.; Zhao, S.; Wang, S.; Liu, D.; Liu, R.P.; Cheng, B.; Chen, J. Automated Policy Combination for Secure Data Sharing in Cross-Organizational Collaborations. *IEEE Access* **2016**, *4*, 3454–3468. [CrossRef]
22. Haguouche, S.; Jarir, Z. Generic Access Control Model and Semantic Mapping Between Heterogeneous Policies. *Int. J. Technol. Diffus.* **2018**, *9*, 52–65. [CrossRef]
23. Ioannidis, S. Security Policy Consistency and Distributed Evaluation in Heterogeneous Environments. Ph.D. Thesis, University of Pennsylvania, Philadelphia, PA, USA, 2005.
24. Mazzoleni, P.; Crispo, B.; Sivasubramanian, S.; Bertino, E. XACML Policy Integration Algorithms. *ACM Trans. Inf. Syst. Secur.* **2008**, *11*, 1–29. [CrossRef]
25. Rao, P.; Lin, D.; Bertino, E.; Li, N.; Lobo, J. Fine-grained integration of access control policies. *Comput. Secur.* **2011**, *30*, 91–107. [CrossRef]
26. Shafiq, B.; Joshi, J.B.D.; Bertino, E.; Ghafoor, A. Secure interoperation in a multidomain environment employing RBAC policies. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 1557–1577. [CrossRef]
27. Ferrini, R. EXAMS: An Analysis Tool for Multidomain Policy Sets. Ph.D. Thesis, University of Bologna, Bologna, Italy, 20 March 2009.
28. Kalam, A.A.E.; Deswarte, Y.; Baina, A.; Kaaniche, M. Access Control for Collaborative Systems: A Web Services based Approach. In Proceedings of the IEEE International Conference on Web Services (ICWS), Salt Lake City, UT, USA, 9–13 July 2007; pp. 1064–1071.
29. Lin, D.; Rao, P.; Ferrini, R.; Bertino, E.; Lobo, J. A Similarity Measure for Comparing XACML Policies. *IEEE Trans. Knowl. Data Eng.* **2012**, *25*, 1946–1959. [CrossRef]
30. Lin, D.; Rao, P.; Bertino, E.; Lobo, J. An Approach to Evaluate Policy Similarity. In Proceedings of the 12th ACM symposium on Access Control Models and Technologies (SACMAT), Sophia Antipolis, France, 20–22 June 2007; pp. 1–10.
31. Ahmadi, S.; Nassiri, M.; Rezvani, M. XACBench: A XACML policy benchmark. *Soft Comput.* **2020**, *24*, 16081–16096. [CrossRef]
32. Deng, F.; Zhang, L.; Zhang, C.; Ban, H.; Wan, C.; Shi, M.; Chen, C.; Zhang, E. Establishment of rule dictionary for efficient XACML policy management. *Knowl. Based Syst.* **2019**, *175*, 26–35. [CrossRef]
33. Deng, F.; Wang, S.; Zhang, L.; Wei, X.; Yu, J. Establishment of attribute bitmaps for efficient XACML policy evaluation. *Knowl. Based Syst.* **2018**, *143*, 93–101. [CrossRef]
34. Dıaz-Lopez, D.; Dolera-Tormo, G.; Gomez-Marmol, F.; Martınez-Perez, G. Managing XACML Systems in Distributed Environments through Meta-Policies. *Comput. Secur.* **2015**, *48*, 92–115. [CrossRef]
35. Li, Y.; Deng, F. A Graph and Clustering-Based Framework for Efficient XACML Policy Evaluation. *Int. J. Coop. Inf. Syst.* **2020**, *29*, 1–17. [CrossRef]
36. Marfia, F.; Neri, M.A.; Pellegrini, F.; Colombetti, M. Using OWL Reasoning for Evaluating XACML Policies. In Proceedings of the International Conference on E-Business and Telecommunications, Colmar, France, 20–22 July 2015; pp. 343–363.
37. Mourad, A.; Tout, H.; Talhi, C.; Otrok, H.; Yahyaoui, H. From model-driven specification to design-level set-based analysis of XACML policies. *Comput. Electr. Eng.* **2016**, *52*, 65–79. [CrossRef]
38. Mourad, A.; Jebbaoui, H. SBA-XACML: Set-based Approach Providing Efficient Policy Decision Process for Accessing Web Services. *Expert Syst. Appl.* **2014**, *42*, 165–178. [CrossRef]
39. Skandhakumar, N.; Reid, J.; Salim, F.; Dawson, E. A policy model for access control using building information models. *Int. J. Crit. Infrastruct. Prot.* **2018**, *23*, 1–10. [CrossRef]
40. Turkmen, F.; Hartog, J.D.; Ranise, S.; Zannone, N. Formal analysis of XACML policies using SMT. *Comput. Secur.* **2017**, *66*, 185–203. [CrossRef]
41. Shvaiko, P.; Euzenat, J. A Survey of Schema-based Matching Approaches. *J. Data Semant. IV* **2005**, *3730*, 146–171.
42. Kuang, T.P.; Ibrahim, H.; Sidi, F.; Udzir, N.I. Heterogeneity XACML Policy Evaluation Engine. In Proceedings of the Malaysian National Conference of Databases (MaNCoD), Selangor, Malaysia, 17 September 2014; pp. 230–238.
43. Do, H.-H.; Melnik, S.; Rahm, E. Comparison of Schema Matching Evaluations. In Proceedings of the Web, Web-Services, and Database Systems, Erfurt, Germany, 7–10 October 2003; pp. 221–237.
44. Liu, L.; Zhang, S.; Diao, L.; Cao, C. An Iterative Method of Extracting Chinese ISA Relations for Ontology Learning. *J. Comput.* **2010**, *5*, 870–877. [CrossRef]

45. Mohan, A.; Blough, D.M. An Attribute-based Authorization Policy Framework with Dynamic Conflict Resolution. In Proceedings of the 9th Symposium on Identity and Trust on the Internet (IDTRUST), Gaithersburg, MD, USA, 13–15 April 2010; pp. 37–50.
46. Miller, G.A. WordNet: A Lexical Database for English. *J. Commun. ACM* **1995**, *38*, 39–41. [CrossRef]
47. Sabou, M.; Lopez, V.; Motta, E. Ontology Selection for the Real Semantic Web: How to Cover the Queen's Birthday Dinner? In Proceedings of the 15th International Conference on Managing Knowledge in a World of Networks (EKAW'06), Podêbrady, Czech Republic, 2–6 October 2006; pp. 96–111.
48. Deng, F.; Zhan, L.Y. Elimination of Policy Conflict to Improve the PDP Evaluation Performance. *J. Netw. Comput. Appl.* **2017**, *80*, 45–57. [CrossRef]
49. Stoller, S.D.; Yang, P.; Ramakrishnan, C.R.; Gofman, M.I. Efficient Policy Analysis for Administrative Role Based Access Control. In Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS), Alexandria, VA, USA, 31 October–2 November 2007; pp. 445–455.
50. Martin, E.; Xie, T.; Yu, T. Defining and Measuring Policy Coverage in Testing Access Control Policies. In Proceedings of the International Conference on Information and Communications Security (ICICS), Raleigh, NC, USA, 4–7 December 2006; pp. 139–158.