Semantic-Based Multi-Domain Data Access Authorization

Noor AbdulKadhim*, Mouiad AL-Wahah*

*Dept. of Computer Science-College of Education-Pure Science-Thi Qar University.
Emails: nourabd7@gmail.com, mouiad-al@utq.edu.iq

Abstract. Different domains employ variety of access control rules to protect the resources lay within their perimeters. When trying to access a resource from outside one’s domain, various issues are arising which prevent cooperating among those domains without endangering the security of the protected resources. The main challenge is how to efficiently handle the rights of users throughout the period of interoperation between various domains. In this paper, we propose a semantic-based multi-domain authorization approach that protects the resources on the multiple domains and, at the same time, provides a steady, flexible and secure authorized access to the protected resources. Two algorithms are described, one for single-domain authorization and the other is for multi-domain authorization. Our approach is based on dynamically merging access control policy rules for various domains in one large ontology, then extracting the access decision. We develop a proof-of-concept implementation and give the complexity analysis for our approach.

Keywords: Multi-Domain (MD), Authorization, Semantic Web (SW), Web Ontology Language (OWL).

1. Introduction

Information exchange and interoperation with one another are essential requirements of emerging systems. Unauthorized access, however, is a major security problem for enterprise application systems in multi-domain environments. The main challenge is how to efficiently handle the rights of users throughout the period of interoperation between various domains. This situation arises when a user at a specific domain requests an access to a resource in another domain [1].

The domain consists of a collection of data, users, and a security policy for governing the links between them. Domain members can have various security features, such as the ability to read, write, or execute permissions on information objects. A domain is at least defined by a domain name, a domain authority name and domain qualification [2]. Multi-domain systems are administrative domains that are autonomous. Several resources exist in each domain that can be shared in large-scale applications between domains. Authorization policies about the resources of a domain are specified by the partnership of one administrator or several administrators [3].
Currently, collaboration among multi-domain systems is a challenge since in addition to heterogeneity issues among different domain policies that must be addressed, it must also be ensured that collaborations are handled securely and that security breaches are effectively monitored and prevented during the secure interoperation process [4].

In order to access information stored on other domains, or to communicate with users on other domains, multi-domain authorization and authentication are required for approving such access operations. Multi-domain authorization is intended to allow domains to share information securely that can be used for authorization [5]. It is possible to view the multi-domain systems as consisting of multiple administrative domains, thus reinforcing knowledge sharing and cooperation across domains, but it is important to concentrate on how to ensure the protection of Web Services in each domain.

However, most of the current access control models are designed for a single local domain and have not considered contact with users and applications in foreign domains, as they can be very different from each other in identifying and developing policies and roles in various domains. This situation has significantly impeded cooperation between different domains and is becoming a major issue of enabling access control of authorized users [6]. The conventional access control models, however, are ideal for predefined resource access management and are manually adapted to policy changes, but that is time-consuming and error-prone [7]. The inter-domain authorization administration lacks adequate protection and flexibility in a multi-domain environment [8].

We present a motivating scenario that explains the need to a robust multi-domain authorization scheme.

“Suppose that a judge is working on a crime case. The judge’s working place is the court which belongs to Justice Ministry; hence the court represents the local domain of the judge’s job. She/he needs to access resources in her/his private domain, but this access is restricted by the judge's security level, the security level of the files needed to be accessed and the judge’s context. This type of access is easy to implement since it is within one domain, the court in our case. However, to reach to a fair decision, the judge may need to access other information resources that are outside her/his private domain. For example, the judge may need to access the accused person’s files, such as DNA prints, fingerprints, investigation sessions, …etc., but these resources lie within another different domain. In this case, these resources are stored at detective office in the police department which belongs to the Ministry of Interior.”

Different information contained in this scenario and most of them are extremely dynamic. The access control policies for the two domains are different, which type of access to be granted? (is it Read? Write?, Reproduce?…etc.), the access control policy in the police department might be very restrictive, and hence prevents the judge from accessing the needed resources or they might be so forgiving and, in this case, endangers the security of the information in the police department and the privacy of the personnel.

To thwart the problems mentioned so far, we propose a multi-domain authorization approach. The approach uses Semantic Web (SW) technologies to provide a facility that allows multi-domain access control. For each domain, we developed an OWL ontology. This ontology serves as the domain definitions and it contains any domain-related specifications. Moreover, the access control policy within each domain’s perimeter is also included in this ontology. However, this access control policy is only useful inside the domain itself and cannot be used to access resources in other domains. For this purpose, a general container ontology is dedicated. The containing ontology serves three objectives:
• It provides a mechanism that overcomes naming convention differences,
• It permits uniting access control policies of the multiple domains,
• It offers the capability of conflict-resolution among different access control policies.

The rest of this paper is structured as follows. Section 2 is dedicated to present background information about Semantic Web technologies and the work that is related to our approach. In section 3, we present our formal representation of Semantic-Based Multi-Domain Data Access Authorization. Section 4 is reserved for showing our implementation results, whereas section 5 draws conclusions and forecasts some future works.

2. Semantic Web Background and Related Work

2.1. Semantic Web Background

The term “Semantic Web” refers to W3C’s vision of the Web of linked data. Semantic Web technologies enable people and/or machines to create data stores on the Web, build vocabularies, and write rules for handling data. Linked data are empowered by technologies such as RDF, SPARQL, OWL, SWRL..., etc. [9].

2.1.1. Resource Description Framework (RDF)

RDF is a simple language used to create standard data models to refer resources, their relationships and data interchange on the web. RDF is a fundamental standard for the Semantic Web. The RDF schema (RDFS) is just an RDF with additional capabilities [10].

2.1.2. Web Ontology Language (OWL)

The OWL 1 language was released as a W3C recommendation on 10 February 2004, while OWL 2 was announced on 27 October 2009. RDF is a generic framework for connecting meta-information to resources. The RDF schema is a collection of RDF resources to be used for describing lightweight ontologies. The OWL language continues this line it can be considered an RDF-based ontology language suitable for writing heavy weight on ontologist. W3C recommends OWL for those cases where the capabilities of the RDF schema are inadequate [11].

2.1.3. Semantic Web Rule Language (SWRL)

The need to reasoning rules on the top of OWL ontologies that are generated by safety, interoperability or collaboration constrains was the reason behind devising the Semantic Web Rule Language (SWRL). SWRL is an ontological language based on OWL and which allows adding rules expression to the ontologies [12].

2.1.4. SPARQL (SPARQL Protocol and RDF Query Language)

SPARQL is a protocol and query language for semantic web data sources. SPARQL can be used to express queries across diverse data sources, whether the data is stored natively as RDF or viewed as RDF via middleware [10].

2.2. Related Work
This section is meant to aid in bringing to light a number of possible issues that already arose with multi-domain authorization approaches developed by researchers and to address the limitations associated with these issues.

Calero et al [13] suggested an authorization model that has advanced features, such as hierarchical RBAC, conditional RBAC, object hierarchies, and task separation, are properly supported. In their research, AL-Wahah and Farkas [14] devise context-based access control authorization framework, an approach suitable for integrating dynamically changing criteria for access control. The specifications for authorization and contextual information are represented as ontologies. The decision on access authorization is focused on the nature of the request and the resources. However, their approach was directed to be used in single-domain environments. Sastry et al [15] develop a modeling paradigm for dynamic authorization in multi-domain systems. However, they neither provide proof for implementing their paradigm nor introduce a proof-of-concept complexity. Li et al [16] suggested a policy framework for executing the privilege query in generic hybrid role hierarchies for specific external requests based on the Minimal Unique Set (MUS). Role mapping is the common method for interoperation between multiple single domains. Role mappings between external and local domains will be calculated based on the outcome of the query. Currently the MUS computation algorithm is non-polynomial. Liu and Huang [17] have proposed the model of Role-Based Access Control Model for Distributed Cooperation Environment RBAC-DC. Despite that, they do not consider the breach of transitivity of domain-to-domain access permissions and their approach has limitations in specifying where the cooperation dependence exists. Kamath et al [18] proposed policy integration technique to integrate multi-domain RBAC. This framework uses roles-associated user-credentials as the key criterion for mapping inter-domain functions. The major problem in user-credential mapping is attributing to fall in labeling conflicts [19].

3. Semantic-Based Multi-Domain Data Access Authorization

In this section, we clarify our approach and how the semantic-based authorization can be implemented efficiently in multi-domain systems. In addition, new policies based on the relationship between access control entities in each domain can be inferred and the access rights needed to satisfy a defined access control policy can be specified.

An authorization system that is designed to work within a single domain so that the user is allowed or prohibited from accessing a specific resource is done according to particular rules that govern access in one domain. In semantic-based world, the authorization requirements are represented within each domain using ontologies as the ones shown in Figures (1) and (2). For example, to allow the judge (who represents the subject in our motivating example given above) to access the files that are with her/his domain, the access control policy rules defined in this domain are enough to enable this kind of access. Let’s suppose that the judge is within domain ddn2, then an OWL ontology is designed to represent the knowledge base of this domain and a set of SWRL rules are put into place (above the OWL ontology) to represent the access control policy rules. The same thing applies for a detective that is working at another domain, say domain ddn3 (refer to Figures (1) and (2)).
The ddn2 ontology defines thirteen basic concepts. These concepts are Allowed, Context, Decision, Denied, Environment, Level, Location, Object, Owner, Request, Role, Type and Subject. The class Subject represents a person or a device that is requesting access to system resources. The class Location has 4 instances, and each refers to a specific location. For example, we have court, Investigation office, policeDept and street. The class Type represents the access type requested by subject. It has 4 instances: delete, copy, write and read. The class Denied all devices/subjects which are denied access to system resources.
Class Owner represents an owner of the information. Class Level represents the security level of the subject and the resource within the same domain, so only the information that is allowed to be viewed by the subject is exposed, and not all of the information, and thus it follows a principle of least privilege concept. It contains 5 subclasses, each of which expresses a special security level (R, C, S, TS, and P). R for restricted access, C for confidential, S for secret, TS for top secret and finally P for public. The class Request represents a request to access system resources, while class Object represents the resource that is to be accessed. Each Object has its own security level. The class Role represents the role played by the subject requesting access.

The authorization process is achieved according to a set of access control policy rules which are encoded using SWRL rules that are specific for each domain and may be very different from other domains. If contextual information is engaged in the authorization process, it is first the case that the authorization engine be able to extract the context of the subject to derive the access decision. For instance, the context of the judge while at business hours is given by SWRL rule R1 shown below.

\[
\text{ddn2: Subject}(x) \land \text{ddn2: withRole}(x, \text{judge}) \land \text{ddn2: atLocation}(x, \text{court}) \rightarrow \text{ddn2: C1}(x)
\]

(SWRL rule R1. Context rule.

We rely on a dynamic way of working that makes sure to work and give a definitive answer, is it authorized or not. The class Context has three subclasses (C1, C2, and C3); each of them determines the temporal or/and spatial context of the access request or the subject requesting access. After running the reasoner (in our implementation, it is Pellet reasoner). For our current example, it will determine that the context C1 will have subjects1, and this is done through the SWRL rule R1 shown above, which includes a set of conditions specifies the characteristics of the subject who is permitted to access.

The class Allowed indicates the decision that can be reached within the same range after running the reasoner. For our current example, decision d1 is allowed, and it is done through the SWRL rule R2, which includes a set of conditions. Running the rules by requesting access and evaluating during runtime decision making, and as shown in rule R2 written in SWRL):

\[
\text{ddn2: Request}(rq) \land \text{ddn2: hasSubject}(rq, x) \land \text{ddn2: C1}(x) \land \text{ddn2: hasID}(x, id) \land \text{ddn2: withPassword}(x, pw) \land \text{ddn2: withLevel}(x, o) \land \text{ddn2: hasType}(rq, t) \land \text{ddn2: onObject}(rq, o) \land \text{ddn2: withValue}(rq, \text{"Accused File"}) \land \text{ddn2: withLevel}(o, l) \land \text{ddn2: Decision}(d) \land \text{ddn2: relatedTo}(d, rq) \rightarrow \text{ddn2: Allowed}(d)
\]

(SWRL rule R2. Allowance rule in domain ddn2.

Depending on the context, the object, and the security level equivalence between the subject and the object, the Judge is authorized in its domain.

The ontology of the second domain ddn3 defines thirteen basic concepts: These concepts are Allowed, Context, Decision, Denied, Environment, Level, Location, Resource, Owner, Request,
Role, Type and Subject. The class Level It represents the security level of the subject and the resources within the same domain. It contains 5 subclasses, each of which expresses a special security level: subclass (Bounded, Classified, General, Private, and Secret), each level has its own access value. The class Resource that represents the information required to be accessed and contains 5 instances and each Object has its own security level. The class Role represents the role played by the subject requesting access. After running the reasoner, the outputs of this class are all subjects that requesting access for resources, and according to our example, The role to subjects requesting resources res1 and res3, the res1 represent data property (Accused File) with level security (Secret) and Judge need to access it, and the res3 represent data property (Crime Fingerprint) with level security (Bounded) and Detective need to access it. The class Decision represents the access decision, as it contains 3 instances, each of which is related to the request. After running the reasoner, the later will determine if this decision is allowed or not. For our current example, d2is also allowed and is associated with the subject s2(Detective) who has some specific context. The class Context also has three subclasses (C1, C2, and C3); each of them determines the temporal/spatial context of the access request or the subject requesting access. Context of subject s2 is dynamically determined using the SWRL rule R3.

\[
\text{ddn3:withRole(?r, ?d), ddn3:atLocation(\text{?x}\, \text{, ddn3:investigationoffice}) \quad \text{ddn3:Subject(?x)}
\]

\[
\wedge \text{ddn3:withValue(?r, "detective") } \rightarrow \text{ddn3:C2(?x)}
\]

(3)

SWRL rule R3. Subject s2 context.

The class Allowed indicates the decision that can be reached within the same domain. For our current example, it is determined by SWRL rule R4.

\[
\text{ddn3:hasID(\text{?x}, ?id), ddn3:Request(?rq, ?d), ddn3:hasType(?rq, ?t), ddn3:Decision(?d), ddn3:related_r(?rq, ?d), ddn3:withLevel(?x, ?l), ddn3:onResource(?rq, ?o), ddn3:hasSubject(?rq, ?x)
}\]

\[
\wedge \text{ddn3:C1(?x)} \wedge \text{ddn3:withValue(?o, "Accused_File")} \wedge \text{ddn3:withLevel(?o, ?l)} \wedge \text{ddn3:withPassword(?x, ?pw)} \rightarrow \text{ddn3:Allowed(?d)}
\]

(4)

SWRL rule R4. Allowance rule in domain ddn3.

Algorithm-1 (which is partially borrowed from M. AL-Wahah and C. Farkas [14]) is used to achieve single-domain authorization. Note that each request is evaluated when it is received by Single-Domain Authorization (SDA) engine.

Algorithm 1 Single Domain Authorization.

Input: A single domain ontology dd_i and req is an access request. R_i is SWRL rule – set Output: Access decision, either "Unauthorized" or "Authorized".

1: D = decompose(req) \ /note that req = (< s, res, t >

2: if t = access then

3: sc = Context(s);

4: d = reason_over(sc, res, dd_i, R_i);

5: if d = "Authorized" AND conflict_free(sc, res, dd_i, R_i) then

6: return("Authorized");
The single-domain authorization algorithm receives a request \( req \), decomposes it to its components, namely the subject \( s \), the resource \( res \) and the access type \( t \). Depending on the access type, the algorithm extracts the context \( sc \) of the subject \( s \) and then derives the suitable decision to be made for this request. The authorization is not going to be permitted until algorithm-1 makes sure that there is no conflict in making the decision. On the contrary, the decision is turned into unauthorized.

However, algorithm-1 is only used within a single-domain authorization and does not support authorization across multiple environments. For this reason, we aim to develop an approach that supports such kind of authorization and the this is achieved using algorithm-2 shown below.

To organize the process of accessing a resource in one domain by a subject in another domain, the subjects must have the same level of security. This means that the judge has the same access rights that the Detective has or is close in terms of level or resource to which she/he is requesting access. The granting of the right of access is for a temporary period, which is the access period, and this is called authorization, but the Detective must be authorized to access within his domain because the decision is copied from the Detective to the judge, so that the judge has access to all the resources in her/his domain, but not all the ddn3 resources, and this is called least privilege. The authorization requirements for multi-domain will be represented by using the ontology shown in Figure (3).
The Multi-Domain Authorization (MDA) ontology, shown above, defines four basic concepts. These concepts are Authorized, Delegatable, Can_Delegate, Domain. The class Domain is concerned with the domains that are trying to communicate with each other. According to our current example, it has two instances D1 which refer to first domain ddn2 and D2 refer to second domain ddn3. The class Delegatable means that the decision can be delegated from one domain to another. The class Can_Delegate says that the administrator responsible for the network, can authorize the access decision or not. After running the reasoner, the outputs are two subjects s1 and s2 which means that the two subjects are equivalent in security level. Hence, s2 can delegate s1, this is done via SWRL rule M1.

\[
\text{SWRL rule M1. Delegation ability checking.}
\]

\[
\begin{align*}
\text{ddn3.Subject(?x1) & ddn3.with(?x2,?y1,?z1): has_level(?x1,?y1,?z1) & ddn1.Subject} \\
\text{(?x2,?y2,?z2): has_level(?x2,?y2,?z2) & greaterThanOrEqual} \\
\text{(?v2,?v1) & con:agreement_EXISTS(?do1,?do2) & con:Can_Delegate(?x1) & con:can_Delegate(?x1,?x2)}
\end{align*}
\]

The class Authorized in MDA ontology represents all subjects who are authorized to access resources which belong to other (or the second) domains via delegation. After running the reasoner, the output is that s1 is authorized in her/his domain as well as being authorized to access resource res3 (in the second domain) and this is done through the SWRL rule M2.

\[
\begin{align*}
\text{ddn3.Request(?rqs1) & ddn3.hasType(?rqs1,?t) & ddn3.hasSubject(?rqs1,?x1)} \\
\text{ddn3.onResource(?rqs1,?o1) & ddn3.Decision(?d) & ddn3.related(?d,?rqs1)} \\
\text{ddn3.allowed(?d) & ddn2.Subject(?x2) & con:needs_to_reach(?x2,?o1) & con:Can_Delegate(?x1,?x2) & con:Authorized(?x2)}
\end{align*}
\]

SWRL rule M2. Authorization rights’ delegation.
Algorithm 2 Multi – Domain Authorization.

Input: Ontologies $dd_1 \ldots dd_n$ and req is an access request. $R_1 \ldots R_n$ are SWRL rule sets, where $R_1 \ldots R_n$ are associated with $dd_1 \ldots dd_n$. MDA ontology and MDAR are multi-domain ontology and multi-domain rule set, respectively.

Output: Access decision, either "Unauthorized" or "Authorized".

1: $D = \text{decompose}(\text{req})$ /* note that $\text{req} = \langle s, \text{res}, t \rangle$ */
2: $\text{dom}_i = \text{domain}(s)$;
3: $\text{dom}_j = \text{domain}(\text{res})$;
4: if $\text{dom}_i == \text{dom}_j$ then
5: call Algorithm_1($dd_i, \text{req}, R_i$);
6: else
7: $\text{merge}(dd_1 \ldots dd_n, \text{MDA}, R_1 \ldots R_n, \text{MDAR})$
8: if $t = \text{access}$ then
9: $sc = \text{Context}(s)$;
10: $d = \text{reason_over}(sc, \text{res}, \text{MDA}, \text{MDAR})$;
11: if $d$ = "Authorized" AND conflict_free(sc, res, MDA, MDAR) then
12: return("Authorized");
13: exit();
14: end if
15: else
16: return("Unauthorized");
17: exit();
18: end if
19: end {Algorithm_2}

Access authorization for a resource shall be granted only if the context of the subject, the context of the resource, and/or any other context specified by the policy on access control corresponds to the particular context of the allowed context. Furthermore, if the main policy engine needs to understand the identification of the subject and resource, it queries its KB store to know that. For example, if we need to know who is the subject within specific context and which domain it belongs to and what request can access resources in another domain; it can simply issue the SPARQL (Language of SPARQL Protocol and RDF Query) query shown in Figure(4) to its KB store:
Figure 4. SPARQL query over main

SPARQL variable is preceded a string by? symbol, so they? Domain, Subject,? Level,?
Request, and ?Resource are variables representing the Domain of subject , the Subject to request
access , the Level of subject, the Resource for resource in second domain. The results of the
SPARQL query are given in table (1) shown below:

Table (1). SPARQL query results.

| ?Domain | ?Subject | ?Level | ?Request | ?Resource |
|---------|---------|--------|---------|-----------|
| D1      | s1      | secret | rq1     | res3      |
4. Implementation Results

Five groups of data are randomly generated and processed to remove duplicates using an online data generator. The teocalli plug-in of Protégée is used to assert these instances to the ontologies of the model. The sets are 1000, 2000, 3000, and 4000 axioms in size. We conducted tests on the MDA engine by submitting SPARQL queries to the populated samples of primary ontologies in our application program. The engine successfully returned only the approved data specified by the sample access control rules. The reasoning process is completed by using the DL reasoner Pellet. DL reasoning has been used to check the accuracy and satisfactoriness of ontology, in addition to the logical subsumption and instance realization. In this paper, the reasoning process is done via Pellet reasoner in the main Java application. For each of the specified data sets, the process of reasoning has been repeated ten times. Then, as shown in table 2, the average time measured for these ten repetitions is taken to obtain specific values for the execution time of the reasoning process. The results of Description Logic (DL) reasoning time are represented in Figure(5). The complexity of the MDA ontology reasoning, which represents the higher order complex ontology in work, is $O(exp)$.

![Figure 5. DL reasoning time.](image)

**Table (2).** Description Logic (DL) reasoning time.

| Axioms | Pellet Reasoner |
|--------|----------------|
| 345    | 187.5          |
| 1345   | 141.5          |
| 2345   | 353.5          |
| 3345   | 585.5          |
| 4345   | 379.6          |
5. Conclusion and Future Work

Cooperation and interoperation across multiple domains are very essential elements in modern networks’ structure. Lack or shortage in multi-domain authorization approaches has led us to propose an algorithm to be used for multi-domain authorization. Our approach is based on dynamically merging access control policy rules for various domains in one large ontology, then extracting the access decision. We develop a proof-of-concept implementation and give the complexity analysis for our approach.

For future work, we are planning to add another OWL ontology to be used as a reference ontology. This ontology, supplied by its own reasoning rules, will serve as a common backup for solving collisions within the multi-domain authorization processes. This will increase the security of the protected resources in the multi-domain environments.

References

[1]. Li, Ruixuan, et al. "Request-driven role mapping framework for secure interoperation in multi-domain environments." Computer Systems Science and Engineering 23.3 (2008): 193.
[2]. Blobel, Bernd, et al. "Modelling privilege management and access control." International Journal of Medical Informatics 75.8 (2006): 597-623.
[3]. Iranmanesh, Zeinab, Morteza Amini, and Rasool Jalili. "A Logic for Multi-domain Authorization Considering Administrators." 2008 IEEE Workshop on Policies for Distributed Systems and Networks. IEEE, 2008.
[4]. Gouglidis, Antonios, Ioannis Mavridis, and Vincent C. Hu. "Security policy verification for multi-domains in cloud systems." International Journal of Information Security 13.2 (2014): 97-111.
[5]. Edwards, Nigel John, and Jason Rouault. "Multi-domain authorization and authentication." U.S. Patent No. 7,444,666. 28 Oct. 2008.
[6]. Wang, Lanjing, and Baoyi Wang. "Attribute-based access control model for web services in multi-domain environment." 2010 International Conference on Management and Service Science. IEEE, 2010.
[7]. Sun, Yuqing, et al. "Active Authorization Management for Multi-domain Cooperation." 2007 11th International Conference on Computer Supported Cooperative Work in Design. IEEE, 2007.
[8]. Cai, Ting, and Jun-Zhan Wang. "MIRBAC: A Role-Based Access Control Model for Multi-Domain Interoperability." International Journal of Security and Its Applications 11.6 (2017): 1-17.
[9]. P. Hitzler, M. Krötzsch, and S. Rudolph, Foundations of semantic web technologies, Chapman and Hall/CRC Press, 2010.
[10]. Farhan Husain M., Doshi P., Khan L., Thuraisingham B. (2009) Storage and Retrieval of Large RDF Graph Using Hadoop and MapReduce. In: Jaatun M.G., Zhao G., Rong C. (eds) Cloud Computing. CloudCom 2009. Lecture Notes in Computer Science, vol 5931. Springer, Berlin, Heidelberg.
[11]. Szeredi, P., Lukácsy, G., Benkő, T., & Nagy, Z. (2014). The Web Ontology Language. In The Semantic Web Explained: The Technology and Mathematics behind Web 3.0 (pp. 407-450). Cambridge: Cambridge University Press.
[12]. O’Connor M. et al. Supporting Rule System Interoperability on the Semantic Web with SWRL. In: Gil Y., Motta E., Benjamins V.R., Musen M.A. (eds) The Semantic Web – ISWC 2005. ISWC 2005. Lecture Notes in Computer Science, vol 3729. Springer, Berlin, Heidelberg.

[13]. Calero, JM Alcaraz, G. Martinez Perez, and AF Gomez Skarmeta. "Towards an authorisation model for distributed systems based on the Semantic Web." IET information security 4.4 (2010): 411-421.

[14]. Mouiad AL-Wahah and CsillaFarkas. "Semantic-Based Authorization: The Need for TheContext." Published in International Journal for Digital Society (IJDS). (2019).

[15]. Sastry Manoj and Ram Krishnan. "A new modeling paradigm for dynamic authorization in multi-domain systems." International Conference on Mathematical Methods, Models, and Architectures for Computer Network Security. Springer, Berlin, Heidelberg, 2007.

[16]. Li, Ruixuan, et al. "Request-driven role mapping framework for secure interoperation in multi-domain environments." Computer Systems Science and Engineering 23.3 (2008): 193.

[17]. Liu, Songyun, and Hejiao Huang. "Role-based access control for distributed COOPERATION environment." 2009 International Conference on Computational Intelligence and Security. Vol. 2. IEEE, 2009.

[18]. Kamath, Ajith, Ramiro Liscano, and Abdulmotaleb El-Saddik. "User-credential based role mapping in multi-domain environment." PST. 2006.

[19]. T. A. Al-Asdi and A. J. Obaid, "An Efficient Web Usage Mining Algorithm Based on Log File Data," Journal of Theoretical and Applied Information Technology, vol. 92, no. 2, pp. 215-224, 2016.