Policy Mediation to Enable Collaborative Use of Sensitive Data

Mathew Cherian
MIT CSAIL
32 Vassar Street
Cambridge, MA
mcherian@MIT.EDU

Lalana Kagal
MIT CSAIL
32 Vassar Street
Cambridge, MA
lkagal@csail.mit.edu

Eric Prud'hommeaux
MIT CSAIL
32 Vassar Street
Cambridge, MA
ericp@csail.mit.edu

ABSTRACT

In the past two decades, industry-academia collaboration has emerged as the new paradigm in pharmaceutical research. The long term success of such partnerships depends on unfettered sharing of data between researchers who operate in two vastly different environments. Moreover, some of these sources contain data that needs to be secured for a variety of reasons: intellectual property, competitive advantage, privacy implications etc. Environments that are able to dynamically integrate data from disparate sources can facilitate aforementioned collaborations. We are developing a secure SPARQL federation engine that can provide such information mash-ups, including data from secured data sources via policy enforcement. In this paper we describe the architecture of our system and discuss its capabilities and contributions.

Categories and Subject Descriptors

H.4.m [**Distributed Systems**]: Miscellaneous; D.2 [**Semantic Web**]

General Terms

Keywords

Secure SPARQL Federation, Proof based authentication, Pharmaceutical research

INTRODUCTION

The integration of data from distributed, heterogeneous data sources is an essential component of realizing the full potential of the Semantic Web - a distributed model for sharing and interpreting information. The development of SPARQL [16], a query language for Resource Description Framework (RDF), has made it easier than before to perform such data integrations. The existence of SPARQL also means that there is an increased incentive for providers to make content available in RDF, a more descriptive framework than traditional relational formats. These advances lead to the need for environments that can perform dynamic integration of data from disparate SPARQL endpoints. Currently, however, there is a lack of secure systems that can perform on-the-fly mash-ups of sensitive data, the access to which needs to be regulated for various legal and economic reasons.

In this paper, we describe the architecture of a data integration engine that provides secure SPARQL federation. The engine accepts a set of queries from a client, sends them off to appropriate SPARQL endpoints, and returns the results to the client. It also requires clients to provide some security credentials, which are used to satisfy the access policies of those endpoints that contain secure data. Such data can include e.g. personally identifiable information(PII) of individuals and trade secrets. For obvious reasons, Linked Open Data (LOD) is not a suitable model for these types of data.

The motivation for the engine's conception, which is described in section 2, is the case of pharmacological research for which collaborative use of data is essential. This work builds on the research done on SPARQL orchestration for open data sources, query rewriting, policy languages and proof generation, which are briefly described in Section 3. In section 4, the architecture of this system is described in detail. In section 5, a summary is provided followed by possible ways to build on this work.

MOTIVATING SCENARIO

In the last decade, industry-academia-health provider partnerships have emerged as the paradigm for pharmaceutical research [5, 8, 14]. Unlike in the past, academics and clinicians are involved in almost every stage of the research process. In such an environment, collaboration between individuals operating in separate domains is essential to the development of new molecules or combinations of existing ones.

When information sharing only takes place between individuals of a particular sector, it is easy to facilitate such sharing. In such a setting, individuals involved in the process have common understandings and expectations for the research processes, rules for interactions with others within the sector, and handling of the end products. For example, in the case of academic research, the resulting work often becomes part of the public domain through publications and/or conferences. As a result, the academic culture is very open and has a propensity to share information, after publication. On the other hand, pharmaceutical companies have a strong economic incentive to protect intellectual property and disclose research data only on a need to know basis. Employees of such organizations are aware of the stakes and as a result, key ideas do not flow freely. Finally, healthcare providers have legal obligations to protect the privacy of their patients and to ensure that PII is not divulged in the process. Such policies are often second nature to those

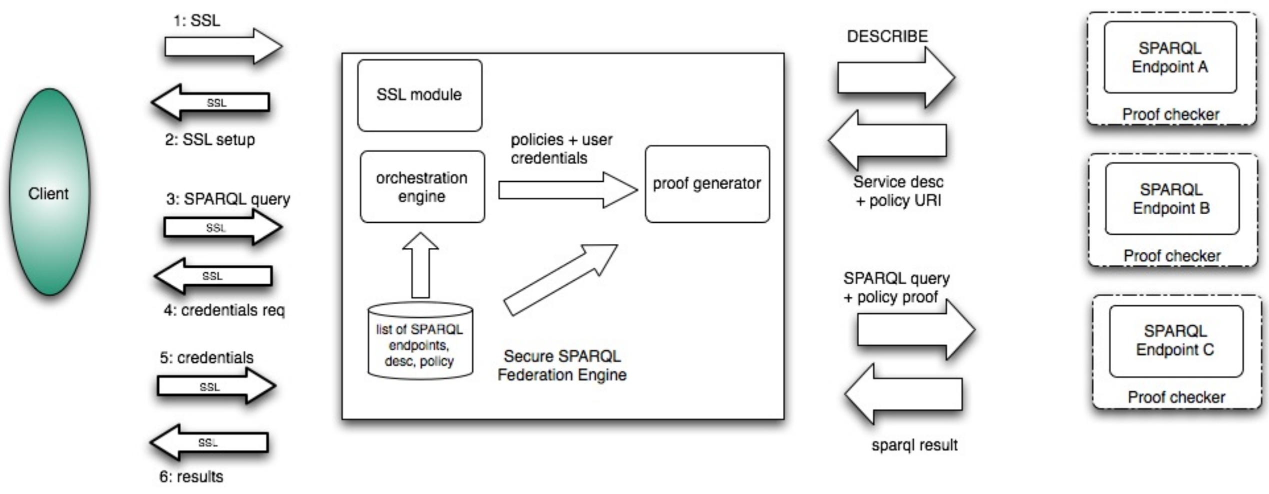


Figure 1: System Architecture

that work in the healthcare sector, but not to others.

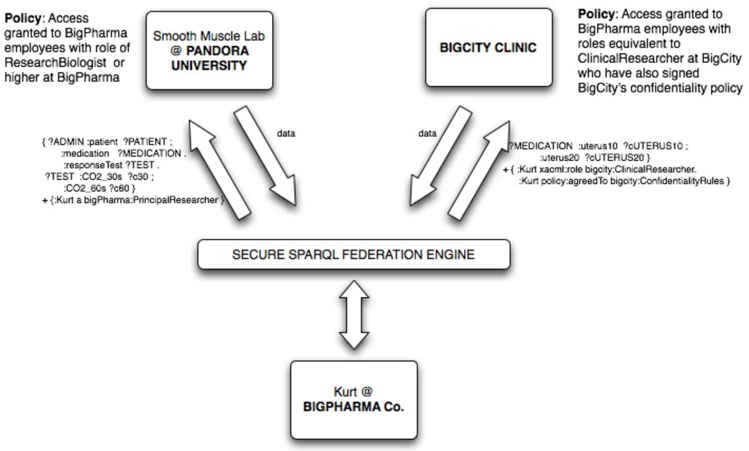
It is easy to see how collaborations that involve parties from these three sectors can easily get bogged down by the differences in bureaucratic structures of each. However, each player brings to the research process unique perspectives and expertise that are invaluable to drug discovery and development. A use case elucidates this scenario more fully.

Kurt conducts genomic research at BigPharma - a pharmaceutical company. Kurt's research is part of a collaborative effort between BigPharma and University of Pandora, which is not geographically close to BigPharma's offices. Most of the research work associated with the project was done at a lab at the university and as such the data is stored on a server on campus. This significant work has produced detailed smooth-muscle responses to 2 agonists in different organs, which is stored behind a secure SPARQL endpoint. Moreover, this research collaboration is a part of a larger effort by BigPharma to develop new therapies for diseases of the lungs.

For this bigger endeavor, BigPharma has established a partnership with BigCity hospital. The purpose of this relationship is to obtain the responsiveness of patients with certain disease expressions to particular medications. BigPharma believes that this information is critical to designing novel drugs based on the newly minted organ response research. BigCity hospital has stored this data in a secure server the access to which is limited to only those healthcare providers that are directly involved in the care of the patients. As part of the collaboration with BigPharma, however, BigCity hospital has agreed to give individuals fulfilling a certain role at BigPharma access to the database. Any access from employees at BigPharma, however, is subject to certain conditions set out in the collaboration agreement between the pharmaceutical company and the hospital. This includes, but is not limited to, removing PII from the accessed files. Employees wishing to access BigCity's data must sign this pre-negotiated confidentiality policy. The policies that dictate BigPharma access are part of the service description of BigCity's endpoint.

Assume that Kurt is looking to identify subjects for a study to investigate whether certain B2 agonists affect the lungs more than the uterus, indicating they would be a better choice for pregnant women. In this case, he needs to query across both SPARQL endpoints and meet their security requirements. A secure federation engine that can integrate data from these two databases would streamline the research process and cut drug development time significantly. Figure 2 shows the different actors and their interactions in our usecase.

We have conceived such a system, which is described in the next sections. We also provide a specific example to illustrate these concepts.



Policy: Access granted to BigPharma employees with role of ResearchBiologist or higher at BigPharma
Smooth Muscle Lab @ PANDORA UNIVERSITY
BIGCITY CLINIC
Policy: Access granted to BigPharma employees with roles equivalent to ClinicalResearcher at BigCity who have also signed BigCity's confidentiality policy
data
data
{ ?ADMIN :patient ?PATIENT ; :medication ?MEDICATION . :responseTest ?TEST . ?TEST :CO2_30s ?c30 ; :CO2_60s ?c60 }
+ {:Kurt a bigPharma:PrincipalResearcher }
?MEDICATION :uterus10 ?cUTERUS10 ; :uterus20 ?cUTERUS20 }
+ { :Kurt xacml:role bigcity:ClinicalResearcher. :Kurt policy:agreedTo bigcity:ConfidentialityRules }
SECURE SPARQL FEDERATION ENGINE
Kurt @ BIGPHARMA Co.

Figure 2: UseCase: Kurt of BigPharma wants to integrate data from the secure SPARQL endpoints of BigCity Clinic and the University of Pandora

RELATED WORK

The research on federated database systems dates back to the 1980s [15]. Much of the work since then has focused on relational databases. Some of this also includes work on secure relational federations. HERMES (HEterogeneous Reasoning and MEdiator System) developed at the University of Maryland was a platform that was developed to design and implement such systems [4].

There has been little work done on such environments for Semantic Web, especially secure ones, due to the relative newness of such technologies. Some work has been done on the semantic integration of relational data using SPARQL [18]. It involves translating SPARQL queries to Datalog to perform all steps involved in data integration optimization, query rewriting, and query optimization. However, it is not capable of querying SPARQL endpoints. Without such queries, much information will lack expressivity and the full potential of the Semantic Web will remain untapped.

The only known federated system that uses SPARQL to query RDF data sources that we are aware of is the DARQ. It is a full-fledged engine that performs query parsing, query planning, query optimization, and query execution. It adapts much of the research on federation of relational databases to perform SPARQL queries on RDF sources. However, DARQ only operates on open data sources and does not offer any support for secure SPARQL federation [17]. As mentioned before, secure data sources are often necessary in scientific, business, and socio-political fields for economic and legal reasons.

ARCHITECTURE

The architecture of the system is illustrated in Figure 1. Its main components are the i) SSL module, which sets up an SSL tunnel for encrypted communication; ii) the orchestration engine, which performs the querying and data integration; and iii) the proof generator, which generates a proof for each secure SPARQL endpoint based on client supplied credentials and endpoint descriptions. The data in the endpoints are in RDF, which means that query results from multiple endpoints can be easily integrated using common variable bindings. The system functions as follows: A client logs into the engine via an SSL connection. Once an SSL link is established, the client submits one or more SPARQL queries to the engine. The orchestration engine accepts the queries and devises a plan to execute the queries on the various endpoints, based on its knowledge of the endpoints' policy descriptions, which are cached. If the plan requires querying of one or more secure endpoints, the engine prompts the user to supply relevant credentials. These credentials, once obtained, are forwarded to the proof generator module to generate proofs that are satisfactory to the endpoint. The queries and proofs, if applicable, are then forwarded to specific endpoints. Once results are received from endpoints, the engine forwards them to the client.

We have already seen some of the benefits of this architecture in the motivating example section. It will become clear in the following subsections, which explain the architecture in detail, how Kurt's work is streamlined by using this system.

SSL

Our system uses the SSL protocol to communicate with a user that wants to query the orchestration engine. The client contacts the engine through an SSL handshake during which the client provides the engine a certificate with a public key. The SSL module uses the public key to authenticate the user. Once this process is complete, the module notifies the client of its decision. If there was successful authentication, the client is allowed to submit SPARQL queries to the federation engine.

In our example, first, Kurt would establish an SSL connection with BigPharma's server with a certificate, which BigPharma had issued to Kurt out of band. The server then authenticates Kurt based on the public key in the certificate and uses the public key as an identifier for him.

SPARQL Orchestration

The crux of the query processing is done by the orchestration engine. It receives a list of SPARQL queries from a user via the secure channel. The engine utilizes the list of source descriptions available to it to determine which endpoints to direct the queries to. Some of the endpoints may specify through their source descriptions the fact that they are secure and that further user credentials are necessary to access data contained in them. If that is the case, the orchestration engine prompts the user for such credentials. The engine then forwards the credentials, if the user provides them, to the proof generation engine to generate satisfactory proofs (a process described in the next subsection) for the particular endpoint. The engine then forwards the queries, along with proofs if necessary, to the endpoints. In the current implementation, the queries are executed in the order they were presented to the engine by the user. Once the responses are received from the endpoints, the orchestration engine forwards either the full results or a justification for the lack of full results back to the client.

It is necessary to have an orchestration engine because subqueries often share variables. Once a sub-query is executed and variables are bound, these mappings are provided to the subsequent subqueries. This process is iterative, it simplifies the query process and reduces execution time.

In our example, BigCity Hospital has a dataset that relates responsiveness of patients with Chronic Obstructive Pulmonary Disease (COPD), a lung condition, to B2 agonists as measured in blood CO2 levels every 30s after administration. The following is a sample of the data that exist in its server.

	B2 agonists	30s	60s	90s	120s
patientX	albuterol	-8%	-14%	-18%	-17%

The Smooth Muscle lab, located at the University of Pandora, has a dataset that related smooth muscle reactivity to B2 agonists. This data would resemble this:

	bronchial			uterus			liver
	10s	20s	30s	10s	20s	30s	...
albuterol	1%	9%	18%	0%	8%	13%	
levoalbuterol	3%	11%	13%	0%	2%	3%	

Kurt is looking to identify subjects for a study to investigate whether certain B2 agonists affected the lungs more than the uterus, indicating that they would be a better choice for pregnant women. In order to do this, Kurt would send the following queries to the orchestration engine

```
Prefix: BC:  <http://studies.bigcity.org/pulmonary/uris/>
Select ?MEDICATION
Where{ ?ADMIN BC:patient ?PATIENT ;
        BC:medication   ?MEDICATION .
        BC:responseTest ?TEST .
  ?TEST  BC:CO2_30s      ?c30 ;
        BC:CO2_60s      ?c60 }
```

```
Prefix:  SML:  <http://www.pandora.edu/research/
                genomics/smooth-muscle/>
Select ?MEDICATION
{ ?MEDICATION  SML:bronchial10  ?cBRONCHIAL10 ;
               SML:bronchial20  ?cBRONCHIAL20 }
```

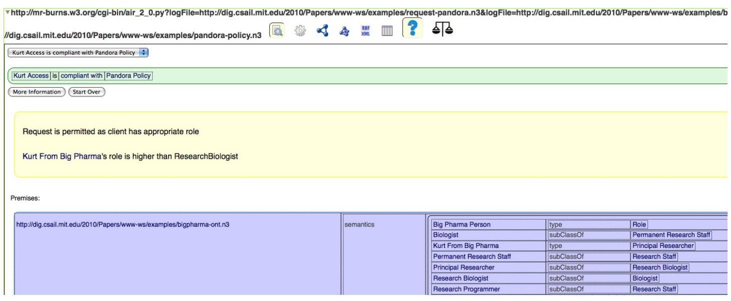
As both SPARQL endpoints are secure, the orchestration engine would then request further credentials, specifying what was requested and which endpoint requested it. This request would be made based on the service descriptions of the endpoint. If Kurt wants to proceed with the query processing, he would provide the requested credentials.

Proof Generation

Our authorization mechanism is based on Proof-carrying Authorization (PCA) [1, 2] and our earlier work on Policy-Aware Web (PAW) [7, 13, 9, 10]. PCA is an authorization framework that is based on a higher-order logic (AF logic) where clients have to generate proofs using a subset of AF logic and the authorizing server's task is to check if the proof is grounded and consistent. This allows objects in the system to have a finer-grained control of the information and enables a smaller trusted computing base on the server. Our work moved these ideas to the open Web using Semantic Web technologies to express policies and proofs.

Though proof generation may be performed by the clients themselves, by delegating it to our system, the load on the client is reduced as are the roundtrips between clients and secure SPARQL endpoints to obtain required credentials. The orchestration engine processes the list of queries input by the client one at a time. When a particular endpoint is found to have a policy, the orchestration engine sends the policy of the endpoint and the client's public key to the proof generator. If unable to generate a proof, the proof generator requests additional credentials based on the policy it is trying to fulfill. The proof generator is a forward chained reasoner [11] that uses client credentials and online resources to generate a proof for how the client meets the specific policy. This proof is returned to the orchestration engine. If the proof generator is not able to generate a required proof based on the client's credentials, the client is informed and has the option to provide additional credentials.

Both SPARQL endpoints in our example are secure and have policies defined in AIR [6], a policy language grounded in Semantic Web technologies. BigPharma has sent BigCity and Pandora its public key offline, so they are able to verify signed statements from BigPharma about its employees, their roles, the projects they work on and other properties. University of Pandora and BigPharma are collaborators and Pandora's policy gives access to those BigPharma employees with role of *Research Biologist* or higher, where the *Research Biologist* and other role terms are defined by BigPharma in RDF. In order to get access to Pandora's endpoint, Kurt must be able to show that he's in a role that is higher than *Research Biologist* and requires a signed statement from BigPharma to prove it. Figure 3 is a screen shot of a proof of Kurt meeting Pandora's policy as viewed in the justification pane of the Tabulator [3]. BigCity Clinic has a slightly more complicated policy. It has mapped BigPharma's role terms to its internal roles (using RDF and Web Ontology Language (OWL)) and restricts access to those individuals who have a similar role to *Clinical Researcher* as defined by them. It also requires BigPharma to inform them if the client has read and signed the confidentiality agreement be-



http://mr-burns.w3.org/cgi-bin/air_2_0.py?logFile=http://dig.csail.mit.edu/2010/Papers/www-ws/examples/request-pandora.n3&logFile=http://dig.csail.mit.edu/2010/Papers/www-ws/examples/b
//dig.csail.mit.edu/2010/Papers/www-ws/examples/pandora-policy.n3

Kurt Access is compliant with Pandora Policy
Kurt Access | is | compliant with | Pandora Policy
More Information
Start Over

Request is permitted as client has appropriate role

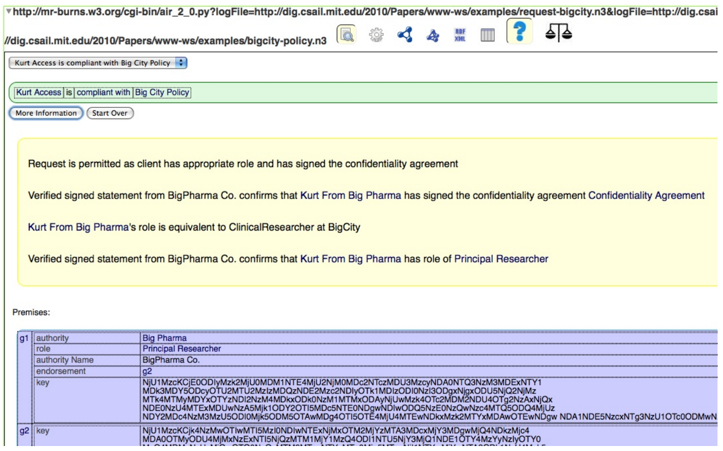
Kurt From Big Pharma's role is higher than ResearchBiologist

Premises:

http://dig.csail.mit.edu/2010/Papers/www-ws/examples/bigpharma-ont.n3 — semantics

Big Pharma Person	type	Role
Biologist	subClassOf	Permanent Research Staff
Kurt From Big Pharma	type	Principal Researcher
Permanent Research Staff	subClassOf	Research Staff
Principal Researcher	subClassOf	Research Biologist
Research Biologist	subClassOf	Biologist
Research Programmer	subClassOf	Research Staff

Figure 3: Part of proof of why Kurt meets Pandora's policy as viewed in Justification UI pane of the Tabulator



http://mr-burns.w3.org/cgi-bin/air_2_0.py?logFile=http://dig.csail.mit.edu/2010/Papers/www-ws/examples/request-bigcity.n3&logFile=http://dig.csai
//dig.csail.mit.edu/2010/Papers/www-ws/examples/bigcity-policy.n3

Kurt Access is compliant with Big City Policy
Kurt Access | is | compliant with | Big City Policy
More Information
Start Over

Request is permitted as client has appropriate role and has signed the confidentiality agreement

Verified signed statement from BigPharma Co. confirms that Kurt From Big Pharma has signed the confidentiality agreement Confidentiality Agreement

Kurt From Big Pharma's role is equivalent to ClinicalResearcher at BigCity

Verified signed statement from BigPharma Co. confirms that Kurt From Big Pharma has role of Principal Researcher

Premises:

g1	authority	Big Pharma
	role	Principal Researcher
	authority Name	BigPharma Co.
	endorsement	g2
	key	NjU1MzcKCjE0ODlyMzk2MjU0MDM1NTE4MjU2NjM0MDc2NTczMDU3MzcyNDA0NTQ3NzM3MDExNTY1 MDk3MDY5ODcyOTU2MTU2MzIzMDQzNDE2Mzc2NDlyOTk1MDIzODI0NzI3ODgxNjgxODU5NjQ2NjMz MTk4MTMyMDYxOTYzNDI2NzM4MDkxODk0NzM1MTMxODAyNjUwMzk4OTc2MDM2NDU4OTg2NzAxNjQx NDE0NzU4MTExMDUwNzA5Mjk1ODY2OTI5MDc5NTE0NDgwNDIwODQ5NzE0NzQwNzc4MTQ5ODQ4MjUz NDY2MDc4NzM3MzU5ODI0Mjk5ODM5OTAwMDg4OTI5OTE4MjU4MTEwNDkxMzk2MTYxMDAwOTEwNDgw NDA1NDE5NzcxNTg3NzU1OTc0ODMwN
g2	key	NjU1MzcKCjk4NzMwOTIwMTI5MzI0NDIwNTExNjMxOTM2MjYzMTA3MDcxMjY3MDgwMjQ4NDkzMjc4 MDA0OTMyODU4MjMxNzExNTI5NjQzMTM1MjY1MzQ4ODI1NTU5NjY3MjQ1NDE1OTY4MzYyNzlyOTY0

Figure 4: Part of proof of why Kurt meets BigCity's policy as viewed in Justification UI pane of the Tabulator

tween BigPharma and BigCity. Kurt needs to have signed statements not only describing his role at BigPharma but also stating whether or not he has agreed to the confidentiality agreement as shown in Figure 4

PAW-enabled SPARQL Endpoint

SPARQL endpoints exist as autonomous entities that interact with the secure orchestration engine via SPARQL queries and responses. Each endpoint may, and ideally will, have some data and corresponding descriptions that are unique to it. An endpoint may also have its own service descriptions that explain its access control and usage policies that suit its needs. The proof generator uses policy descriptions to generate a proof, if necessary, of why a client is allowed access to a particular endpoint. An endpoint may exist as an open endpoint or one that is secure and hence requires some proof from the orchestration engine that verifies a users identity and authorization to obtain the data. Such secure endpoints have a built-in proof checking component [13], which verifies the proofs. The orchestration engine contacts the endpoints periodically to update its cache of each SPARQL endpoints data and service descriptions. These updates need not take place simultaneously.

BEYOND ACCESS CONTROL

Our current architecture focuses on providing upfront authorization mechanisms while integrating data from secure SPARQL endpoints. However, several policies such as the ones mentioned in our motivating use case - not releasing non-aggregate information, ensuring appropriate anonymization, and restricting use to research - cannot be enforced a priori. We propose the use of accountability mechanisms as a means of ensuring appropriate use of information. We view accountability combined with transparency and appropriate redress as a complementary process to strict access control [19]. This approach requires extensive and system-wide audit trails, the ability to express usage policies and automated reasoning engines that interpret these policies to determine whether the particular uses of data in the logs are policy-compliant. Towards this end, we plan to annotate all results of SPARQL endpoints with usage policies that provide the client with a machine-understandable representation of what they are/are not permitted to do with the data. The existence of such policies would ensure that individuals are not identified as a result of PII that may result from mashing up individual datasets, which by themselves do not contain any PII.

SUMMARY

We have presented the architecture for a first of its kind secure SPARQL orchestration engine. It provides a secure channel using the SSL protocol and accepts a list of queries from the user. It then presents the queries to appropriate SPARQL endpoint that have registered with it ahead of time. This process is facilitated by the source, data and policy descriptions, which the engine initially received during registration and updated periodically. If an endpoint only permits secure access, the engine provides it with a proof, generated from the credentials, along with the query. The enpoints policy descriptions, which the query engine has cached, are used in generating the proof. This system was motivated by and has major applications in the area of pharmaceutical research. It facilitates easy access to scientific data across different domains universities, private sector research centers, and healthcare providers, while ensuring that appropriate safeguards are in place to allow only authorized access to data.

We are aware that the provision of multiple queries by a client is a limitation of our system. A query optimization algorithm is being developed, which if implemented would permit full scale SPARQL federation. For this work, we plan to rely on an algorithm developed as part of a project for query rewriting on the semantic web [12]. Once such optimization is integrated, the client would be able to submit a single SPARQL query and get a single answer to that query. We are setting up the design of the rest of the system so that different optimization modules may be plugged in depending on the tasks at hand.

We are in the final stages of finishing the implementation of the orchestration engine. We plan to incorporate it with the proof generator entity, which was the result of our earlier work on Policy Aware Web.

Acknowledgements

The authors would like to thank Bo Andersson and Elgar Pichler (W3C HCLSIG working group) for their help in developing the use case. This work was supported in part by AFOSR YIP award FA9550-09-1-0152 and IARPA award number FA8750-07-2-0031.

REFERENCES

[1] L. Bauer. *Access Control for the Web via Proof-carrying Authorization*. PhD thesis, Princeton University, Nov. 2003.

[2] L. Bauer, S. Garriss, and M. K. Reiter. Distributed proving in access-control systems. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 81–95, May 2005.

[3] T. Berners-Lee, J. Hollenbach, K. Lu, J. Presbrey, E. Prud'ommeaux, and mc schraefel. Tabulator Redux: Browsing and Writing Linked Data . In *Linked Data on the Web Workshop at WWW08*, 2008.

[4] K. Candan, S. Jajodia, and V. Subrahmanian. Secure mediated databases. In *Proceedings of the Twelfth International Conference on Data Engineering*, March 1996.

[5] I. Cockburn. The changing structure of the pharmaceutical industry. *Health Affairs*, January/February 2004.

[6] Decentralized Information Group. Air policy language. http://dig.csail.mit.edu/2009/AIR/, 2009.

[7] Decentralized Information Group and MINDSWAP. Policy-aware web project. http://www.policyawareweb.org/, 2006.

[8] J. Jankowski. Trends in academic research spending, alliances, and commercialization. *Journal of Technology Transfer*, June 1999.

[9] L. Kagal, T. Berners-Lee, D. Connolly, and D. Weitzner. Self-describing delegation networks for the web. In *IEEE POLICY '06: Proceedings of the Seventh IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 205–214, Washington, DC, USA, 2006. IEEE Computer Society.

[10] L. Kagal, T. Berners-Lee, D. Connolly, and D. J. Weitzner. Using semantic web technologies for policy management on the web. In *21st National Conference on Artificial Intelligence*, 2006.

[11] L. Kagal, C. Hanson, and D. Weitzner. Using dependency tracking to provide explanations for policy management. In *IEEE Policy 2008*, 2008.

[12] D. Kolas. Query rewriting for semantic web information integration. In *Sixth International Workshop on Information Integration on the Web*, 2007.

[13] V. Kolovski, Y. Katz, J. Hendler, D. Weitzner, and T. Berners-Lee. Towards a policy-aware web. In *In Semantic Web and Policy Workshop at the 4th International Semantic Web Conference*, 2005.

[14] K. C. London. Largest academic-industry collaboration for drug discovery in depression and schizophrenia launched. January 2010.

[15] D. McLeod and D. Heimbimger. A federated architecture for database systems. In *National Computer Conference*, 1980.

[16] E. Prud'hommeaux and A. Seaborne. Sparql query language for rdf w3c recommendation. http://www.w3.org/TR/rdf-sparql-query, January 2008.

[17] B. Quilitz and U. Leser. Querying distributed rdf data

sources with sparql. In *5th European Semantic Web Conference (ESWC2008)*, pages 524–538, June 2008.

[18] J. Wang, Z. Miao, Y. Zhang, and J. Lu. Semantic integration of relational data using sparql. In *Second International Symposium on Intelligent Information Technology Application*, pages 422–426, 1996.

[19] D. J. Weitzner, H. Abelson, T. Berners-Lee, J. Feigenbaum, J. Hendler, and G. J. Sussman. Information accountability. *Communications of the ACM*, June 2008.