On the Use of Ontology Data for Protecting Critical Infrastructures

J Henriques[1,2], F Caldeira[1,2], T Cruz[1], P Simões[1]

[1]*Department of Informatics Engineering*
University of Coimbra
Coimbra, Portugal

Email: jpmh@dei.uc.pt; fmanuel@dei.uc.pt; tjcruz@dei.uc.pt; psimoes@dei.uc.pt

[2]*Polytechnic Institute of Viseu*
Viseu District, Portugal

Abstract: *Modern societies increasingly depend on products and services provided by Critical Infrastructures (CI). The Security Information and Event Management (SIEM) systems in charge of protecting these CIs usually collect and process data from specialised sources. However, they usually integrate only a small fraction of the whole data sources existing in the CI. Valuable generic data sources are missing in this process, such as human resources databases, staff check clocks, and outsourced service providers. To address this gap, the authors propose a framework that takes a Semantic Web approach for automated collection and processing of corporate data from multiple heterogeneous sources*.

Keywords: *Critical Infrastructure Protection (CIP), Security Information and Event Management (SIEM), Industrial Automation and Control Systems (IACS), Semantic Web, Ontologies*

Introduction

Critical Infrastructures (CI) such as telecommunication networks and power grids are becoming increasingly complex and interdependent on people, processes, technologies, information, and other critical infrastructures. Operators in charge of Critical Infrastructure Protection (CIP) are required to improve their security levels through the perspective of compliance auditing and forensic analysis. Compliance auditing is related to applicable security regulations, standards, and best practices. Forensic analysis has a broader scope, beyond the specific operations of the CI industrial control systems, and also encompasses other areas of the organisation.

The benefits of enlarging the scope of information sources for SIEM applications, forensic analysis, and compliance audit operations are rather evident, since the result would enable more powerful, all-inclusive approaches to cybersecurity awareness. For example, monitoring of abnormal activity within the IACS specific domain might be leveraged by the correlation of different data sources, such as mail filtering logs (monitoring phishing and malware attacks, which target the employees of the CI) and information about employee functions residing in Human Resources information systems. Another example would be the correlation of data from physical access control systems and staff check clocks with activity logs of IACS operators. In general, this strategy of associating core security information already fed into SIEM systems with peripheral-awareness data would result in richer security analysis processes that enable

the detection of inconsistencies, malpractices, and intrusion clues, which would otherwise go unnoticed.

However, achieving tight integration of all those peripheral data sources into the already-existing SIEM frameworks is costly and often impractical. This would require considerable investments in data conversion and adaptation to the SIEM data flows. Moreover, the maintenance costs would also be considerable, since even minor adjustments on the corporate information systems would require explicit adaptations on the SIEM side.

A more plausible option is, therefore, the adoption of loosely coupled integration strategies, such as resorting to Semantic Web approaches for automating the processing and interpretation of large amounts of information available from both local databases and Internet repositories. This reasoning process, applied over a large quantity of available data with knowledge inferred from a combination of axioms, properties, and rules (with different levels of hierarchies or categorisations and deriving conclusions, for example) can be explicitly expressed by ontologies.

It should be noted that most data are still not directly available in Semantic Web formats. This is the case with data maintained in Relational Databases (RDBs). Nonetheless, mapping data from RDB to Semantic-Web-enabled Resource Description Frameworks (RDFs) has been the focus of a large body of previous research, leading to the implementation of many generic mapping tools and their applications, on several specific domains. Those tools are natural candidates to be adapted to the field of CIP so that security-related ontology data currently stored in heterogeneous databases can be taken into consideration—despite the considerable challenges involved, such as the migration from existent systems to the semantic level (Sernadela, González-Castro & Oliveira 2017).

A detailed discussion of the main motivations and driving research efforts in mapping RDB to RDF can be found in Sahoo *et al.* (2009). Although most models can perform inference from native ontology data stores, data still reside mostly in RDBs, which are broadly used within organisations. Moreover, the growing number of datasets published on the Web brings opportunities for extensive data availability and challenges related to the process of querying data in a semantically heterogeneous and distributed environment. The structured query approach fails on the linked data because the Web's scale makes it impractical for users to know in advance the structure of datasets (Freitas, *et al*. 2012).

The authors have previously introduced an approach considering inference capabilities from Semantic Web, supported by common schemas, for creating a set of independent databases, each deployed with its own domain-specific schema (Henriques *et al.* 2018). This kind of reasoning is suitable for application in the context of Critical Infrastructure Protection; and, therefore, it can leverage current SIEM capabilities—mainly in what relates to forensic and compliance audit processes, but also for intrusion detection purposes. This large amount of living heterogeneous data that still resides in the organisational RDBs will, in this way, become available to the Critical Infrastructure's SIEM and enable new, valuable insights into available configuration and monitoring data.

This paper refines and extends previous work (Henriques *et al.* 2018) by providing a more detailed description of the proposed approach, adding a practical use-case scenario, and discussing its future application to different data sources.

After discussing some of the key previous work and trends in the area, this paper takes a practical approach by presenting the implementation of a federated query architecture for retrieving a set of compliance auditing rules that might be useful, for instance, in assessing CI security levels. To leverage inference capabilities, it maps the living data currently available on RDBs into RDFs formats. In this way, it can substantially enlarge the data available to the SIEM by taking advantage of the large amount of heterogeneous data of production-RDB systems. Such an approach provides an abstraction mechanism for keeping data consumers away from low-level details while leveraging the security concerns of the underlying infrastructures by hiding the internal deployment aspects, such as the identification of the involved machines and their RDB schemas.

The ontology-based approach of this work considers the available information currently stored in RDB and, as its main goal, makes it accessible through simple interfaces that collect queried data from multiple natively different data repositories within the organisation. Each available RDB maintains different information instances, deployed on specific schemas and technologies. Such an approach is suitable for combining data from two different worlds, such as the case of RDB and Semantic Web data, which is natively maintained in RDF stores and made available through an interface layer encapsulating the details of the gathering process to retrieve the data from multiple RDBs.

The remainder of this paper is structured as follows. The next section discusses the background for the domain problem and related work. Immediately following is an analysis of the applicability of ontology data in the context of CIP. Next, a description of the proposed architecture, which details its implementation, will be provided. Finally, the authors conclude the paper with insights about future developments.

Background

This section briefly introduces the reader to the key concepts and tools used in the proposed data integration approach: RDF; RDB, and RDF mapping; SPARQL; Direct Mapping of Relational Data to RDF; and the D2RQ platform.

Resource Description Framework (RDF)

An ontology is a formal specification of concepts (Gruber 1993) in a domain of discourse, which includes classes and properties. An ontology, together with a set of individual instances of classes, constitutes a knowledge base (Noy & McGuinness 2001).

The Resource Description Framework (RDF) (Brickley & Guha 1999) is a language that can be used to encode knowledge into web pages to make them understandable for electronic agents searching for information. This is one of the main goals for using ontologies (Musen 1992; Gruber 1993). RDF aims at representing information that may be used for inference purposes over the Web. The RDF syntax core structure consists of a set of triples with a subject, a predicate, and an object. A set of triples is called an RDF graph. An RDF graph may be visualised as a directed-arc diagram, in which each triple is represented as a node-arc-node link. RDF is a data format based on a Web-scalable architecture for identification and interpretation of terms (RDF 2014).

Mapping from RDF to RDB

As already mentioned, the mapping of large amounts of data from RDB to RDF has been the focus of intense research work in multiple domains and has led to the implementation of a set of generic mapping tools, as well as domain specific applications. RDF has provided an

integration platform for data gathered from multiple sources, primarily from RDB. This is one of the main motivations driving research efforts (using various approaches) on mapping RDB to RDF (Seaborne, *et al*. 2013).

SPARQL (W3 2013) can be used to express queries across diverse data sources, whether for data natively stored as RDF or for data viewed as RDF via some sort of middleware. SPARQL is a World Wide Web Consortium (W3C) recommendation for querying multiple RDF graphs. The SPARQL specifications define the syntax and semantics to proceed with queries across diverse natively stored RDF data sources. Using the latest stable release (1.1), SPARQL federated queries allow merging multiple results retrieved from multiple RDF sources. The syntax and semantics of SPARQL 1.1 Federated Query extension allow distributed queries over different SPARQL endpoints. Moreover, the SERVICE clause extends SPARQL 1.1 to support queries that merge data distributed across the Web. A single query is, therefore, able to return related data (for example, contacts to be applied to user John Doe) from multiple distinct SPARQL endpoints.

An important feature of RDF and SPARQL is that they can use different datasets from different locations, federating them together. They offer a middleware, which can use multiple data sources as if they were one. Moreover, it is simple to add and remove data sources. This feature significantly reduces the development costs as compared to typical data warehouse projects (DuCharme 2013).

Figure 1 provides a query example through different SPARQL 1.1 endpoints. The query returns John's contacts from two distinct SPARQL endpoints, *www.site1.com* and *www.site2.com*.

```
SELECT ?contact1
WHERE {
SERVICE <http://www.site1.com/sparql>
 {SELECT ?contact1
WHERE {
       ?me foaf:nick "John".
       ?me foaf:knows ?f .
       ?f foaf:name ?contact1
       }
}
SERVICE <http://www. site2.com/sparql>
 {
SELECT ?contact2
WHERE {
       ?me foaf:nick " John ".
       ?me foaf:knows ?f .
       ?f foaf:name ?contact2 }}
       FILTER (?contact1 = ?contact2)
       }
}
}
```

Figure 1: Query example through different SPARQL 1.1 endpoints

Direct mapping of relational data to RDF

Relational databases allow the use of tools, such as Structured Query Language (SQL), for accessing and managing the databases. Several strategies already exist to map relational data to RDF. Typically, the goal is to describe the RDB contents using an RDF graph, allowing

queries submitted to the RDF schema to indirectly retrieve the data stored in relational databases. A direct mapping process enables a simple transformation and can be used for materialising RDF graphs or for defining virtual graphs, which can be queried via SPARQL or traversed by an RDF graph Application Programming Interface (API). A mapping document is an RDF document containing triples maps with instructions on how to convert relational database content into RDF graphs.

The D2RQ platform

The D2RQ (Data to RDF Query) Platform allows users to access relational databases as virtual, read-only RDF graphs while automatically producing the corresponding mappings. It is available under the Apache open source license (D2RQ 2012), and it allows users to create customised mappings from RDB through an integrated environment with multiple options for accessing relational data, including RDF dumps, Jena and Sesame API based access, and SPARQL endpoints on D2RQ Server (Bizer & Cyganiak 2007). It offers RDF-based access to the content of RDB, without requiring its replication into RDF stores. D2RQ, therefore, allows querying non-RDF databases using SPARQL or accessing contents of databases over the Web. It also allows the creation of custom content dumps from relational databases into RDF stores.

The D2RQ Platform includes components such as a Mapping Language, an Engine, and a D2R (Data to RDF) Server. The D2RQ Engine is a plug-in for the Jena Semantic Web toolkit, which uses mappings for rewriting the Jena API calls to SQL queries against the database and for redirecting query results up to the higher layers of the framework. The D2R Server is an HTTP server which provides linked data views, HTML views for debugging, and a SPARQL protocol endpoint providing an interface to query the database. The D2RQ platform supports databases such as MySQL, SQL Server, Oracle, PostgreSQL, HSQLDB, and Interbase/Firebird. Some limitations of D2RQ include the integration of multiple databases or other data sources and its read-only nature: it lacks Create, Read, Update, and Delete (CRUD) operations. Finally, it does not support inference mechanisms and does not include named graphs (D2RQ 2012).

The D2RQ Mapping Language enables defining relationships between RDB schemas and RDF schema vocabularies (classes and properties) or Web Ontology Language (OWL) ontologies written in Turtle syntax (W3 2014). The mapping properties define a virtual RDF graph, which contains information from the database schema. The mapping process between D2RQ and RDB entities includes the RDF class node to RDB tables and RDF predicates to RDB column names (D2RQ 2012).

The same D2RQ server can be configured to access multiple databases. Therefore, a single SPARQL query can request data from multiple databases at once, which is not possible with a standard SQL query.

Applicability of Ontology Data in the Context of Critical Infrastructure Protection

This section addresses the applicability of ontology data in the context of CIP. First, some of the more pertinent related works are discussed. Afterwards, the H2020 ATENA module for forensics and compliance auditing is presented. This module provides the framework on which the proposed approach, described in the following section, was developed.

Related work

Current approaches on the use of ontologies in the context of CIP are mostly related to the assessment of interdependencies between Critical Infrastructures, such as the works of

Castorini *et al.* (2010) and Blackwell *et al.* (2008). Similarly, a proposal for an ontology providing vulnerabilities classification to be used in decision support tools can be found in Chorás *et al*. (2010).

Other approaches worth mentioning include SPLENDID, DARQ, SemaPlorer, and FedX. SPLENDID (Gorlitz & Staab 2011) is a query optimisation strategy for federating SPARQL endpoints based on statistical data. DARQ (Quilitz 2008) provides transparent query access to multiple SPARQL services using one single RDF graph, even when data has a distributed nature and is spread over the Web. This approach includes a service description language that enables a query engine to decompose a query into subqueries, where each of them can be answered by an individual service. SemaPlorer (Schenk *et al.* 2009) also provides a federated query architecture allowing it to interactively explore and visualise semantically heterogeneous distributed semantic datasets in real time, through a conceptual layer on top of Amazon's Elastic Computing Cloud (EC2). FedX (Schwarte, *et al*. 2011) proposes novel joint processing and grouping techniques for minimising the number of remote requests. It also develops a practical framework that enables efficient SPARQL queries supported by federation layers for efficient query processing on heterogeneous distributed Linked Open Data sources.

Beyond D2RQ, other RDF middleware applications exist, such as TopQuadrant's TopBraid Live, OpenLink Software's Virtuoso Sponger, and Triplr project. These offer dynamic creation and integration. They also allow users to merge several RDF triples in a single SPARQL endpoint from sources such as relational databases, spreadsheets, HTML documents, and other formats.

As already mentioned, one possible application of ontology data in this scope is the use of heterogeneous sources available in organisational RDBs for leveraging inference capabilities. This application is especially interesting in the specific areas of forensic analysis and compliance audit processes, which, by nature, need to be supported by substantial amounts of heterogeneous data. A possible practical application of this approach, in the scope of forensic analysis and compliance audit processes, may consist of the collection and mapping to Semantic Web of rules residing in the multiple and heterogeneous relational databases of the CI organisation—so they can be combined with the knowledge already available at the SIEM systems. This path has been explored in the scope of the H2020 ATENA research project (ATENA 2018; Rosa *et al.* 2017), as discussed next.

Forensics and compliance auditing in the scope of the H2020 ATENA framework

The H2020 ATENA project proposes an innovative logical framework, with design improvements of role, operation, architecture, and security components for Industrial Automation and Control Systems (IACS), while also exploiting novel security approaches enabled by network virtualisation paradigms. The Forensics and Compliance Auditing (FCA) module, integrated into the ATENA cyber-security architecture, addresses the gathering and persistent storage of digital evidence retrieved from both the cyber-analysis layer (such as SIEM) and peripheral sources (such as service logs, sessions, or physical access control systems, among others) for forensics and compliance auditing purposes. Its forensics tools provide the means to identify, extract, preserve, and highlight digital evidence for technical investigation and legal purposes. Its compliance auditing tools support the audit procedures associated with certification processes for applicable standards, policies, and regulations—for example, verifying the authorisation procedures for physical installation access, such as access to doors (Rosa *et al.* 2017).

Moreover, the FCA module provides a set of analysis capabilities for interactively exploring, searching, extracting, pinpointing, and combining insights from available data. The core FCA functions encompass collecting heterogeneous data from internal and external sources, producing structured and unstructured data to be combined and gathered into a unified view for compliance auditing—throughout a set of rules—and also providing forensic investigation functionalities for retrieving evidence.

Figure 2, below, depicts the main blocks of the ATENA FCA module. Data collected from the ATENA SIEM and intrusion detection systems feed a specific CI security data lake which provides input to the FCA analytics components. Peripheral data sources, processed through domain-specific business rules, also feed the analytics layer. Trust and repudiation indicators are also used to assess the trustworthiness of each data source.

As previously discussed, specific ontologies need to be constructed for supporting the already mentioned processes of compliance audit and forensics analysis. In the context of the FCA module hereby presented, the targets for the use of those ontologies are the Analytics sub-components 'Audit Compliance' and 'Forensic Analysis'.

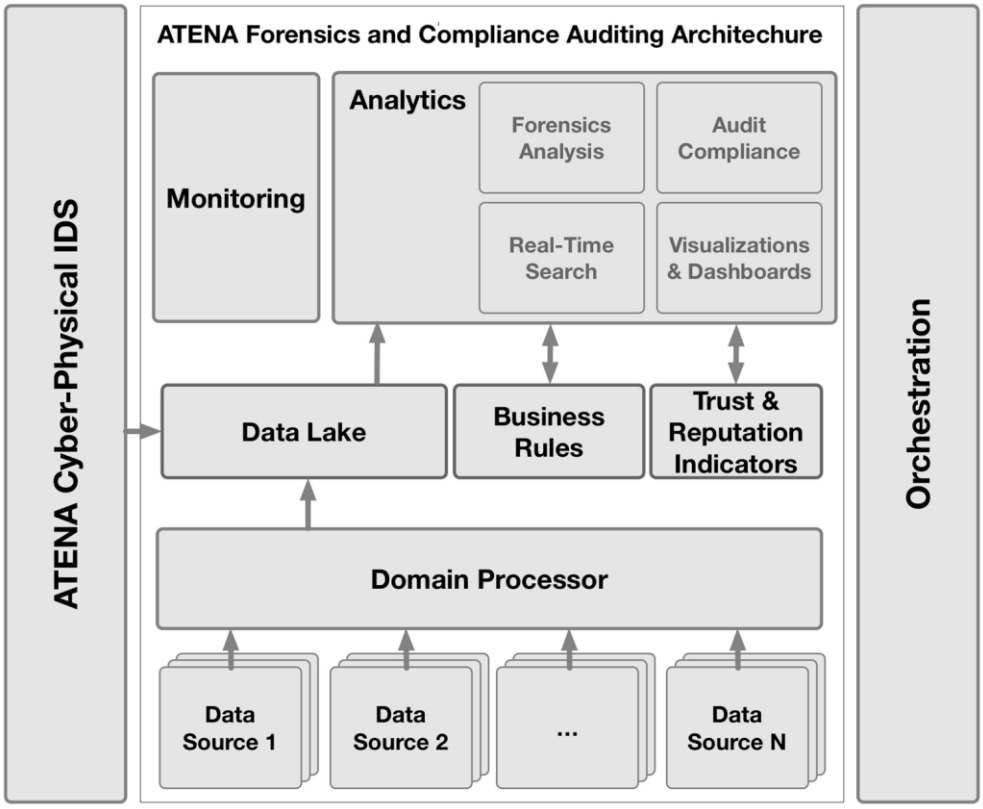

Figure 2: The Forensics and Compliance Auditing Module of the ATENA Project, adapted from Rosa, *et al*. (2017)

Proposed Approach to the Use of Ontology Data for CIP

This section describes the proposed approach to the use of ontology data in the context of CIP applications. First, the proposed reference architecture is introduced, followed by a discussion of technical aspects and implementation details. In a simplified view, the proposed solution consists of a web service that can receive several SPARQL requests from data consumers (such as the forensics and compliance auditing tools mentioned in the previous sections). Afterwards,

each one of those requests is forwarded into different databases deployed using different schemas.

Reference architecture

The proposed reference architecture, depicted in **Figure 3**, below, consists of a set of components such as a federated layer, mapping brokers, and databases. Several data consumers (clients) may send distinct sets of SPARQL queries to the federated interface layer, which delivers each query to all the brokers. The broker's main role is to transform the incoming SPARQL queries into native relational database queries. Through an inverse flow, the broker retrieves the data subset from the database to be gathered into a full data set at the federated layer which is then forwarded to the involved client(s).

Although the reference architecture may suggest its applicability to the context of federated database queries, it may be extended to use different kind of data sources, such as logs or Lightweight Directory Access Protocol (LDAP) distributed directory information services (among others) in order to provide compliance audit and forensic capabilities that can be applied to the context of ATENA FCA module.

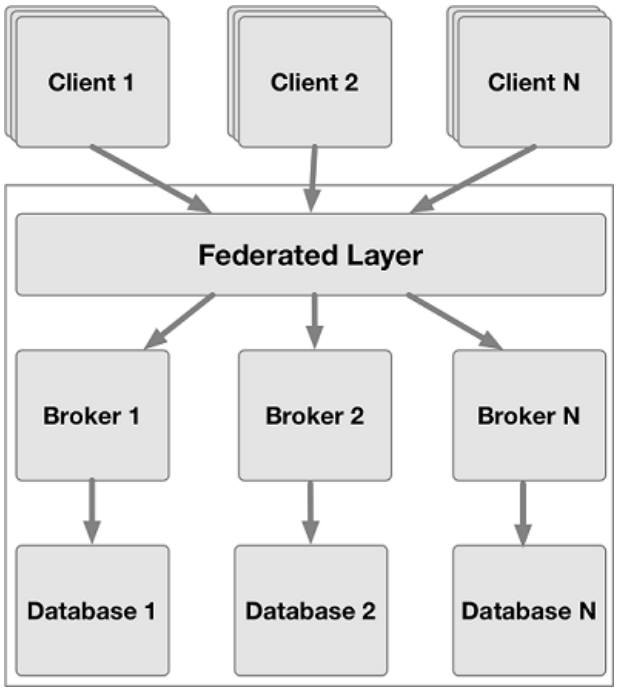

Figure 3: Proposed reference architecture

Use-case scenario

Next, a simple compliance audit scenario is presented, which demonstrates the applicability of the reference architecture for evaluating unauthorised accesses to the assets of an international company.

The challenge is to build a common schema for the management of human and asset resources spread over different platforms, because of specific requirements imposed by national governments. A single interface, capable of answering queries merging all the data in the organisation in a single dataset, should be provided. Such an approach would help overcome the barriers by approaching different native data sources spread across different locations in an organisation.

Implementation aspects

This implementation starts by modelling a simple ontology for the forensic and compliance audit processes, which encompasses the norms, policies, and legal or regulatory guidelines that are being applied. The ontology will allow users to infer new knowledge, for example, to identify possible unauthorised or incompatible access to the assets of a large organisation. This example implements a federated query web service for evaluating whether employees have the required roles when they access those assets. An intermediary layer translates the requests arrived to the web service into queries for the internal schemas of the involved databases.

The interface layer is implemented as a web service, while the mapping brokers are implemented as D2R Server endpoints. Each endpoint is assigned to different relational database(s)**. Figure 4**, below, provides a general overview of the implementation of the described architecture, depicting how requests flow from a submitted query to the web service, which implements a federated query solution to dispatch the incoming requests to the indexed list of database servers—with each of them mapped by a specific D2RQ component. For simplicity's sake, the figure includes just two different databases with different schemas (one Microsoft database—MSSQL—and one MySQL database), but there are no limits to the number or type of involved databases.

The use case hereby described involves a client requesting the contents of the 'Roles' database entity. The objective is to gather and combine—without requiring the end user to be aware of low-level details—information dispersed across different tables and different databases which use different schemas. After the request query to retrieve the existing contents from the 'Rules' entity has reached the database instances, each delivers its contents to a SPARQL endpoint through a D2R server assigned to each involved database. The D2RQ Mapping Language is used for the mapping process. This central web service allows clients to directly query existing entities, to retrieve available content from each existing database, and to merge and deliver them to the querying clients.

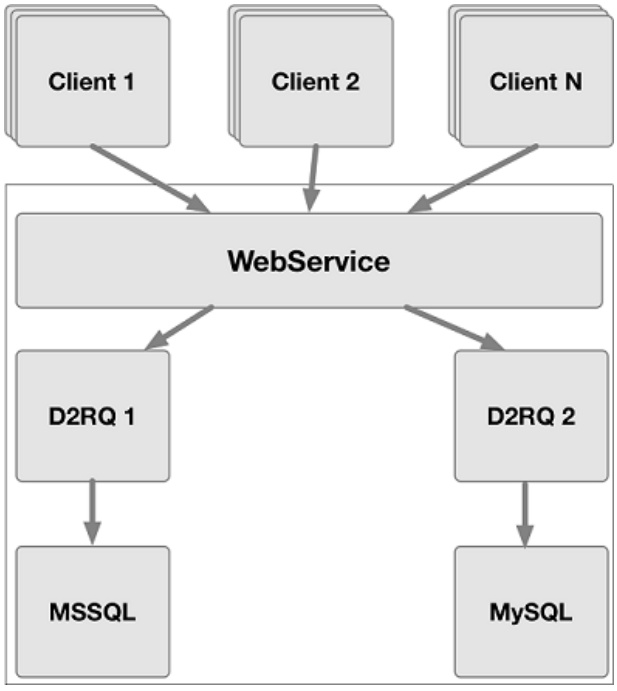

Figure 4: Architecture implementation

Required tools and technologies include Visual Studio as development environment, C# as programming language, ASPX.NET for implementing the web service, classic RDBs such as MSSQL and MySQL, and the RDF and SPARQL languages describing their semantics.

The following sections discuss some details for each step involved in the implementation and deployment of this specific use case. First, a simple ontology is presented. Next, some relevant implementation steps are discussed, such as deploying the database server, generating mapping, configuring the mapping between the database server and the ontology, activating D2R servers with the corresponding mappings, and describing the web service.

Create the Ontology: In this section, a simple ontology is explored in the domain of compliance audit to support the previously presented use-case scenario, which has the main purpose of answering the following question: 'Who is able to access the assets, for maintenance purposes, in a large company spread out through different countries and businesses?'

The ontology, built within Protégé, includes classes for 'Asset', 'Employee', 'Organization', and 'Role'. The corresponding instances are 'Computer', 'John' and 'Francis', 'PowerPlantA', and 'MaintainsIT'. The ontology does not include any hierarchy of concepts.

Table 1, below, summarises the relationship among class instances, their types and property assertions.

Instance	**Type**	**Property Assertions**
John	Employee	**isEmployedBy**:PowerPlantA **Number**: '1002' **Name**: 'John'
Francis	Employee	**isEmployedBy**: PowerPlantA **hasRole:**MaintainsIT **Number**: '1001' **Name**: 'Francis'
MaintainsIT	Role	**maintains**:Computer **isMaintainedBy**: Francis **Name**: 'Francis'
PowerPlantA	Organization	**hasEmployees**:John **hasEmployees**: Francis **hasAssets:** Computer **Name:** 'PowerPlantA'
Computer	Asset	**isRoledBy**: Francis **belongsTo**:PowerPlantA **Number:** '10000001' **Name:** 'DELL'

Table 1: Classes instances

'John' and 'Francis' are instances of 'Employee'. Both have the property 'isEmployedBy' assigned with the value 'PowerPlantA'. The employee is assigned roles granting the access to the assets, enabling the building of a query to assess the regulatory rules and policies. It also has as an inverse property 'hasRole' as 'MaintainsIT'. Additionally, they have data properties '1' and '2' for the 'Number', and 'Francis' and 'John' for 'Name'. Notwithstanding, the difference between 'Francis' and 'John' instances is that the 'Francis' does not include the property 'hasRole' as 'MaintainsIT'. Therefore, they will be considered two employees for the organisation, but just one of them is able to maintain the assets.

‘PowerPlantA’ is an instance of the ‘Organization’ type and includes the property ‘hasEmployees’ for ‘Francis’ and ‘John’ instances. Therefore, this organisation has two employees. ‘Computer’ is an instance of the ‘Asset’ type and its properties are ‘isRoledBy’ of the ‘MaintainsIT’ instance, whose value is ‘Francis’ and which includes a ‘Number’ and a ‘Name’.

Figure 5 provides the full contents of the above ontology, in turtle language, located at ‘data.ttl’ file:

```
#filename: data.ttl
@prefix FCA: <http://www.semanticweb.org/FCA#>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix owl: <http://www.w3.org/2002/07/owl#>
@prefix rdfs: <http: //www.w3.org/2000/01/rdf-schema#>
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>

#################################################################
#    Object Properties
#################################################################

###  http://www.semanticweb.org/FCA#belongsTo
FCA:belongsTo rdf:type owl:ObjectProperty ;
owl:inverseOf FCA:hasAssets ;
rdfs:domain FCA:Asset ;
rdfs:range FCA:Organization .

###  http://www.semanticweb.org/FCA#hasAssets
FCA:hasAssets rdf:type owl:ObjectProperty ;
rdfs:domain FCA:Organization ;
rdfs:range FCA:Asset .

###  http://www.semanticweb.org/FCA#hasEmployees
FCA:hasEmployees rdf:type owl:ObjectProperty ;
owl:inverseOf FCA:isEmployedBy ;
rdfs:domain FCA:Organization ;
rdfs:range FCA:Employee .

###  http://www.semanticweb.org/FCA#hasRole
FCA:hasRole rdf:type owl:ObjectProperty ;
owl:inverseOf FCA:isRoledBy ;
rdfs:domain FCA:Employee ;
rdfs:range FCA:Role .

###  http://www.semanticweb.org/FCA#isEmployedBy
FCA:isEmployedBy rdf:type owl:ObjectProperty ;
rdfs:domain FCA:Employee .

###  http://www.semanticweb.org/FCA#isRoledBy
FCA:isRoledBy rdf:type owl:ObjectProperty ;
owl:inverseOf FCA:isRoledBy ;
rdfs:domain FCA:Role ;
rdfs:range FCA:Employee .

#################################################################
#    Data properties
#################################################################

###  http://www.semanticweb.org/FCA#Name
```

```
FCA:Name rdf:type owl:DatatypeProperty ;
rdfs:domain FCA:Asset ,
FCA:Employee ,
FCA:Organization ,
FCA:Role .

###  http://www.semanticweb.org/FCA#Number
FCA:Number rdf:type owl:DatatypeProperty ;
rdfs:domain FCA:Asset .

#################################################################
#    Classes
#################################################################

###  http://www.semanticweb.org/FCA#Asset
FCA:Asset rdf:type owl:Class .

###  http://www.semanticweb.org/FCA#Employee
FCA:Employee rdf:type owl:Class .

###  http://www.semanticweb.org/FCA#Organization
FCA:Organization rdf:type owl:Class .

###  http://www.semanticweb.org/FCA#Role
FCA:Role rdf:type owl:Class .

#################################################################
#    Individuals
#################################################################

###  http://www.semanticweb.org/FCA#Computer
FCA:Computer rdf:type owl:NamedIndividual ,
FCA:Asset ;
FCA:belongsTo FCA:PowerPlantA ;
FCA:Name "DELL"^^xsd:string ;
FCA:Number "10000001"^^xsd:int .

###  http://www.semanticweb.org/FCA#Francis
FCA:Francis rdf:type owl:NamedIndividual ,
FCA:Employee ;
FCA:hasRole FCA:MaintainsIT ;
FCA:isEmployedBy FCA:PowerPlantA ;
FCA:Name "Francis"^^xsd:string ;
FCA:Number "1001"^^xsd:int .

###  http://www.semanticweb.org/FCA#John
FCA:John rdf:type owl:NamedIndividual ,
FCA:Employee ;
FCA:isEmployedBy FCA:PowerPlantA ;
FCA:Name "John"^^xsd:string ;
FCA:Number "1002"^^xsd:int .

###  http://www.semanticweb.org/FCA#MaintainsIT
FCA:MaintainsIT rdf:type owl:NamedIndividual ,
FCA:Role ;
FCA:isRoledBy FCA:Francis ;
FCA:Name "MaintainsIT"^^xsd:string .

###  http://www.semanticweb.org/FCA#PowerPlantA
```

```
FCA:PowerPlantA rdf:type owl:NamedIndividual ,
FCA:Organization ;
FCA:hasAssets FCA:Computer ;
FCA:hasEmployees FCA:Francis ,
FCA:John ;
FCA:Name "PowerPlantA"^^xsd:string
.
```

Figure 5: Ontology definition

Deploying the database server: This step involves the creation of the table objects for MySQL and MSSQL databases, as well as the commands for populating them. For the sake of demonstration, the MSSQL database table schemas and contents are different from the ones used in the MSSQL database. At the end, these two databases should maintain different data over distinct schemas, which will become federated at the upper level of the web service. The applied commands were the following:

```
generate-mapping -u root -p password01pt -o ssfile_MYSQL.ttl -d
com.microsoft.                                sqlserver.jdbc.SQLServerDriver
jdbc:sqlserver://host_mysql;databaseName=BD_mssqlDB

generate-mapping -u sa –p password02pt -o ssfile_SQLServer.ttl –d
com.microsoft.                                sqlserver.jdbc.SQLServerDriver
jdbc:sqlserver://host_mssql;databaseName=BD_mysqlDB
```

Prepare mapping: The mapping process between database and RDF schemas is mapped through the ‘ssfile_SQLServer.ttl’, whose contents include the mapping between the MSSQL server and RDF schemas—the ‘ssfile_MYSQL.ttl’ file plays the same role, but for the MySQL schema. The initial section of these files includes a set of prefixes (several were removed from the next listing for clarity), with the *map:database* component providing a way for retrieving information from the database server. These files were manually updated to allow the correct mapping between RDF and the database schemas. This mapping is supported by RDF *d2rq:ClassMap* and *d2rq:PropertyBridgefor* classes and properties, respectively. **Figure 6** includes the contents for mapping the class ‘Employee’ and table ‘Employee’ from the MSSQL server:

```
@prefix map: <#> .
@prefix db: <> .
@prefix vocab: <vocab/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix d2rq: <http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ/0.1#> .
@prefix jdbc: <http://d2rq.org/terms/jdbc/> .

map:database a d2rq:Database;
      d2rq:jdbcDriver
"com.microsoft.sqlserver.jdbc.SQLServerDriver";
      d2rq:jdbcDSN
"jdbc:sqlserver://localhost;databaseName=BD_joaohenriques";
      d2rq:username "joaohenriques";
      d2rq:password "password1";
      .

# Table CREATE TABLE dbo.Employee (Number INT, Name VARCHAR(100))
```

```
map:dbo_Employee a d2rq:ClassMap;
       d2rq:dataStorage map:database;
       d2rq:uriPattern "dbo/Employee/@@dbo.Employee.Number@@";
       d2rq:class vocab:dbo_Employee;
       d2rq:classDefinitionLabel "dbo.Employee";
       .
map:dbo_Employee__label a d2rq:PropertyBridge;
       d2rq:belongsToClassMap map:dbo_Employee;
       d2rq:property rdfs:label;
       d2rq:pattern "Employee #@@dbo.Employee @@";
       .
map:dbo_Employee_Number a d2rq:PropertyBridge;
       d2rq:belongsToClassMap map:dbo_Employee;
       d2rq:property vocab:dbo_Employee_Number;
       d2rq:propertyDefinitionLabel "Employee Number";
       d2rq:column "dbo.Employee.Number";
       d2rq:datatype xsd:integer;
       .
map:dbo_Employee_Name a d2rq:PropertyBridge;
       d2rq:belongsToClassMap map:dbo_Employee;
       d2rq:property vocab:dbo_Employee_Name;
       d2rq:propertyDefinitionLabel "Employee Name";
       d2rq:column "dbo.Employee.Name";
       d2rq:datatype xsd:string;
```

Figure 6: Mapping between RDF and database schemas

Activate D2R servers*:* The next step deploys the D2R server, in order to map the contents from RDB to RDF according to the mapping file. The following command activates the MSSQL and MYSQL servers respectively:

```
d2r-server -p 2021 ssfile_SQLSERVER.ttl

d2r-server -p 2020 ssfile_MYSQL.ttl
```

Activate web service: The web service provides the main functions performing the federation mechanism and retrieving the information from the SPARQL endpoints. The web service provides an interface and a federated query layer and offers query services that allow end users to perform the intended inference operations while remaining abstracted from low-level details. Each submitted query is forwarded to multiple RDBs through a DR2Q component. The results are later merged into a single result set. The endpoints are configured at server level, and take into consideration the fact that the end user does not need to know the number or the location of such existing endpoint servers. The web service endpoint is located at

```
http://host_webservice:17129/WebService1.asmx?op=SemanticWEB.
```

Query the ontology: The final step is to query the knowledge base. The SPARQL query in **Figure 7**, below, requests the knowledge base for assessing which users are authorised to execute the maintenance of the assets in a given organisation. This query is forwarded from the Web service to all the federated SPARQL endpoints assigned to different databases and which is finally translated into the internal schema of those databases. The query filters the organisation ‘PowerPlantA’ for the asset ‘Computer’, where just some of the employees having the role ‘MaintainIT’ are authorised to perform its maintenance:

```
PREFIX : <http://www.semanticweb.org/FCA#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT *
WHERE{
       ?employee rdf:type owl:NamedIndividual.
       ?employee :hasRole ?role.
    ?organization rdf:type :Organization.
       ?organization :hasEmployees ?employee.
       ?asset rdf:type owl:NamedIndividual.
       ?role :isRoledBy ?employee.
       FILTER(?organization = :PowerPlantA)
       FILTER(?asset = :Computer )
       FILTER(?role = :MaintainsIT )
 }
```

Figure 7: SPARQL query for assessing authorised users

Figure 8, below, demonstrates the use of the Apache Jena SPARQL command '*sparql --data=data.ttl --query query.rq*' and the corresponding output. The query contents are located in the 'query.rq' file, which was used against data located at the 'data.ttl' file. According to the knowledge base, just 'Francis' is able to execute the 'Computer' maintenance.

```
C:\Users\jpmh\Downloads>sparql  --data=data.ttl --query query.rq
-----------------------------------------------------
| employee | role         | organization | asset     |
=====================================================
| :Francis | :MaintainsIT | :PowerPlantA | :Computer |
-----------------------------------------------------
```

Figure 8: SPARQL command

Discussion and Conclusions

This paper proposes an approach for leveraging inference capabilities in the use of heterogeneous data currently maintained in multiple, natively different RDB systems. This approach aims at contributing to Critical Infrastructure Protection by supporting activities such as forensic analysis and compliance audit procedures. It provides Semantic Web reasoning capabilities through an interface able to answer to federated queries. The process of interactively exploring, searching, extracting, pinpointing, and combining insights can use and combine data sourced from disparate organisational RDBs. Thus, this approach avoids the duplication of information in RDB and RDF stores, and overcomes the issues arising from the use of static data integration (such as the lack of support for transformations of data and the effort required for maintaining up-to-date synchronisation processes). The proposed web service includes an abstraction layer that deals with inherent complexities of resorting to different platforms, systems, technologies, and information schemas to retrieve and to combine heterogeneous data. This abstraction layer also improves security by hiding the infrastructure's internal details.

Although the approach taken by the proposed federated architecture is similar to the one of SPARQL 1.1, it does not require previous knowledge about the existence and location of SPARQL endpoints. The benefits of this approach come from the inclusion of an abstraction layer, which provides direct access to operational data that live in different organisational RDBs. Details such as the involved database servers and differences between schemas can be

kept away from users. Moreover, it is flexible enough for leveraging the exploration of additional data sources that might be easily added in the future. The proposed framework also provides a data fusion solution for gathering multiple data items—representing the same real-world object—into a single, consistent, and clean representation.

This work arises from the limited research on the use of ontology data for CIP applications, and the need to improve and facilitate the usage of the huge amounts of data living in the RDBs of Critical Infrastructure operators. This work also explored Semantic Web inference tools, and is aimed at the practical objective of federating queries against a knowledge base containing the ontology and data for assessing employee authorisations for asset maintenance in a large organisation that uses multiple different RDBs. This practical approach suggests a future path for the improvement of CIP by using inference capabilities for forensic and compliance audit purposes and leveraging the use of heterogeneous ontology data living in RDBs and in other heterogeneous kinds of data sources.

Acknowledgements

This work was partially funded by the ATENA H2020 Project (H2020-DS-2015-1 Project 700581).

References

ATENA 2018 'H2020 ATENA Project website', viewed 24 May 2017, <https://www.atena-h2020.eu/>.

Bizer, C & Cyganiak, R 2007, 'D2RQ: Lessons learned', Position paper for the W3C, Workshop on RDF Access to Relational Databases, Cambridge, MA, US.

Blackwell, J, Tolone, WJ, Lee, SW, Xiang, WN & Marsh, L 2008, 'An ontology-based approach to blind spot revelation in critical infrastructure protection planning', *Proceedings of the International Workshop on Critical Information Infrastructures Security*, Springer, Berlin, Heidelberg, DE, pp. 352-59.

Brickley, D & Guha, RV 1999, *Resource Description Framework (RDF) Schema specification, Proposed recommendation, World Wide Web Consortium*, viewed 12 December 2017, <http://www.w3.org/TR/PR-rdf-schema>.

Castorini, E, Palazzari, P, Tofani, A & Servillo, P 2010, 'Ontological framework to model Critical Infrastructures and their interdependencies', *Proceedings of Complexity in Engineering*, COMPENG'10, pp. 91-3.

Chorás, M, Kozik, R, Flizikowski, A & and Hołubowicz, W 2010, 'Ontology applied in decision support system for critical infrastructures protection', *Proceedings of the international conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2010): Trends in Applied Intelligent Systems*, pp. 671-80.

D2RQ 2012, 'D2RQ', viewed 31 August 2017, <http://d2rq.org>.

DuCharme, B 2013, *Learning SPARQL: Querying and updating with SPARQL 1.1*, 2nd edn, O'Reilly Media, Inc.

Freitas, A, Curry, E, Oliveira, JG, and O'Riain, S 2012, 'Querying heterogeneous datasets on the linked data Web: Challenges, approaches, and trends', *IEEE Internet Computing*, vol. 16, no. 1, pp. 24-33.

Gorlitz, O & Staab, S 2011, 'SPLENDID: Sparql endpoint federation exploiting void descriptions', *Proceedings of the second international conference on Consuming Linked Data*, vol. 782, pp. 13-24.

Gruber, TR 1993, 'A translation approach to portable ontology specification', *Knowledge Acquisition*, vol. 5, pp. 199-220.

Henriques J, Caldeira F, Cruz T & Simões P 2018, 'On the use of ontology data for protecting critical infrastructures', *Proceedings of the 17th European Conference on Cyber Warfare and Security (ECCWS)*, Oslo, NO.

Musen, MA 1992, 'Dimensions of knowledge sharing and reuse', *Computers and Biomedical Research*, vol. 25, pp. 435-67.

Noy, NF & McGuinness, DL 2001, 'Ontology development 101: A guide to creating your first ontology', *Stanford knowledge systems laboratory technical report KSL-01-05*, viewed 23 November 2017, <http://www.corais.org/sites/default/files/ontology_development_101_ aguide_to_creating_your_first_ontology.pdf>.

Quilitz B & Leser U 2008, 'Querying distributed RDF data sources with SPARQL', S Bechhofer M Hauswirth, J Hoffmann, M Koubarakis (eds), *The Semantic Web: Research and applications*, European Semantic Web Conference (ESWC) 2008, Lecture notes in computer science, vol. 5021, Springer, Berlin, Heidelberg, DE, pp. 521-38.

RDF (Resource Description Framework) 2014, 'W3C Resource Description Framework (RDF)', viewed 4 September 2017, <https://www.w3.org/RDF>.

Rosa L, Proença J, Henriques J, Graveto V, Cruz T, Simões P, Caldeira F & Monteiro E 2017, 'An evolved security architecture for distributed industrial automation and control systems', *Proceedings of the 16th European Conference on Cyber Warfare and Security (ECCWS).*

Sahoo, S, Halb, W, Hellmann, S, Idehen, K, Thibodeau, T, Auer, S, Sequeda, J & Ezzat, A 2009, *A survey of current approaches for mapping of relational databases to RDF*, viewed 24 October 2017, <http://www.w3.org/2005/Incubator/rdb2rdf/RDB2RDFSurveyReport.pdf>.

Seaborne A, Polleres A, Feigenbaum L & Williams, G 2013, 'SPARQL 1.1 Federated Query', position paper for the W3C Workshop on SPARQL 1.1 Federated Query, viewed 4 September 2017, <https://www.w3.org/TR/sparql11-federated-query/>.

Schenk, S, Saathoff, C, Staab, S & Scherp, A 2009, 'SemaPlorer—Interactive semantic exploration of data and media based on a federated cloud infrastructure', *Web Semantics: Science, services and agents on the World Wide Web*, vol. 7, no. 4, pp. 298-304, viewed 13 September 2017, <http://doi.org/10.1016/j.websem.2009.09.006>.

Sernadela, P, González-Castro, L & Oliveira, JL 2017, 'SCALEUS: Semantic Web services integration for biomedical applications', *Journal of Medical Systems*, vol. 41, no. 4, p. 54.

Schwarte, A, Haase, P, Hose, K, Schenkel, R & Schmidt, M 2011, 'FedX: Optimization techniques for federated query processing on linked data', L Aroyo, *et al*. (eds), *The Semantic Web – ISWC 2011*, Lecture notes in computer science, vol. 7031, Springer, Berlin, Heidelberg, DE, pp.601-16.

W3 2013, 'SPARQL', viewed 4 September 2017, <https://www.w3.org/TR/sparql11-query>.

——2014, 'SPARQL', viewed 4 September 2017, <https:// www.w3.org/TR/turtle>.