

# PRE4J - A Partial RDF Encryption API for Jena

Mark Giereth

Visualization and Interactive Systems Institute, University of Stuttgart  
Universitätsstraße 38, 70569 Stuttgart, Germany  
giereth@informatik.uni-stuttgart.de

**Abstract.** PRE4J is a cryptographic extension for the Jena Semantic Web Framework. It provides a lightweight API for encrypting sensitive information in an RDF graph while the non-sensitive parts remain publicly readable. The graph holding the sensitive information in an encrypted form is RDF-compliant and self-describing by using an RDF vocabulary. PRE4J uses XML Encryption and XML Signature for the representation of encrypted information and encryption metadata. The API is built on top of the Jena `com.hp.hpl.jena.graph` package.

## 1 Introduction

Real world Semantic Web applications have to be able to handle different kinds of sensitive data. Therefore, methods for specifying which information in an RDF graph is allowed to be processed by which agents are necessary. In general there are two different approaches to achieve this goal. The first is to specify access rights or access policies [15,4] and to control the data access. The second is to use cryptographic methods to protect the sensitive information itself.

However, all access control approaches need trusted infrastructures for checking policies and controlling the data access. A secure way for storing RDF graphs that contain sensitive information in insecure environments (e.g. publicly readable web-spaces or shared desktop systems) is to locally encrypt them before storing them.

The ability to re-use distributed data has been an important design issue for the Semantic Web (c.f. [9]). From the perspective of re-useability encrypting complete RDF graphs is not desirable and a partial encryption method, where only the sensitive information is encrypted while the non-sensitive information remains publicly readable, should be preferred. The idea of Partial RDF Encryption (PRE) has been introduced in [8]. Possible applications for PRE could be:

- Publishing of metadata for selected trusted agents, for example when only trusted crawlers should be able to process the provided data.

- Secure storage of personal data in common RDF stores.

- Group internal information exchange using public Web infrastructures, for example notifications via RSS [7] feeds which are encrypted for the members of a working group.

- Using RDF to 'semantically' manage secure information, like passwords, Transaction Authentication Numbers (TANs), etc.

This paper proposes PRE4J, a cryptographic extension for Jena [2]. PRE4J allows for fine-grained encryption of sensitive information in an RDF graph. Encrypted

and plaintext information is represented in a single RDF-compliant graph after graph transformations on the original graph have been performed. PRE4J uses XML Encryption [5] and XML Signature [6] for representing encrypted information and encryption metadata in XML. PRE4J is built on top of Jena's `com.hp.hpl.jena.graph` System Programmer Interface (SPI) and the Java Cryptography Extension (JCE) [1] to implement the encryption functionalities.

The rest of this paper is organized as follows. In the next two sections an overview over the PRE process is given describing the encryption of RDF graphs, the necessary graph transformations, and the handling of blank nodes. The algorithm for mapping the graph encryption to atomic add and remove operations on triples is briefly described in section 3. Section 4 introduces the PRE4J API and gives an example. The paper ends with a discussion and gives an outlook to future work.

## 2 Partial RDF Encryption

An RDF graph is a set of assertions modeled as  $(s; p; o)$ -triples, where  $s$  (subject) identifies the resource the assertion is about,  $p$  (predicate) identifies a property of the resource, and  $o$  (object) is the value of  $p$ . Each triple is an element of  $T = (U \cup B) \times U \times (U \cup B \cup L)$ , where  $U$  denotes the set of URIs as defined in [3],  $B$  denotes the set of blank node identifiers, and  $L$  denotes the set of RDF literals as defined in [10]. The functions  $\text{subj}(hs; p; oi) = s$  and  $\text{obj}(hs; p; oi) = o$  are defined to return the subject and the object of a triple. Each set of  $(hs; p; oi)$ -triples can be interpreted as a Directed Labeled Graph (DLG) with subjects and objects as labeled nodes, triples as arcs, and predicates as arc labels  $(s, p, o)$ . In general, a DLG encodes two different types of information: structural information and label information. A DLG can be partially encrypted by encrypting selected labels, nodes or arcs. Labels can be encrypted by replacing them with their ciphers. Nodes and arcs can be encrypted by replacing structural information with some kind of 'secure containers' that allow the re-construction of the original DLG structure when the appropriate keys are provided for decryption.

Standard encryption algorithms like DES [12], AES [13] or RSA [14] can be used to encrypt sensitive elements of an RDF graph. Elements can be either subjects, predicates, objects, triples or triple sets. Each element to be encrypted is serialized using a RDF serialization language, such as N-Triples, RDF/XML, N3, etc. The result is a string representation which is used as input for the encryption function.

Let  $f$  and  $g$  be encryption functions with corresponding decryption functions  $f^{-1}$  and  $g^{-1}$ . Each function is parameterized with a key so that  $f_{k_1}^{-1}(f_{k_2}(m)) = m$  holds for each message  $m$ .  $f$  is called symmetric if  $k_1 = k_2$ , else it is called asymmetric. PRE4J uses a combination of symmetric and asymmetric encryption. An overview













