Science of Computer Programming 121 (2016) 153–175

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Tracing known security vulnerabilities in software repositories – A Semantic Web enabled modeling approach

Sultan S. Alqahtani, Ellis E. Eghan, Juergen Rilling *

ARTICLE INFO

Article history:
Received 1 April 2015
Received in revised form 23 January 2016
Accepted 27 January 2016
Available online 4 February 2016

Keywords:
Semantic knowledge modelling
Semantic web
Software security vulnerabilities
Software traceability
Impact analysis

ABSTRACT

The introduction of the Internet has revolutionized not only our society but also transformed the software industry, with knowledge and information sharing becoming a central part of software development processes. The resulting globalization of the software industry has not only increased software reuse, but also introduced new challenges. Among the challenges, arising from the knowledge sharing is Information Security, which has emerged to become a major threat to the software development community, since not only source code but also its vulnerabilities are shared across project boundaries. Developers are unaware of such security vulnerabilities in their projects, often until a vulnerability is either exploited by attackers or made publicly available by independent security advisory databases. In this research, we present a modeling approach, which takes advantage of Semantic Web technologies, to establish traceability links between security advisory repositories and other software repositories. More specifically, we establish a unified ontological representation, which supports bi-directional traceability links between knowledge captured in software build repositories and specialized vulnerability database. These repositories can be considered trusted information silos that are typically not directly linked to other resources, such as source code repositories containing the reported instances of these problems. The novelty of our approach is that it allows us to overcome some of these traditional information silos and transform them into information hubs, which promote sharing of knowledge across repository boundaries. We conducted several experiments to illustrate the applicability of our approach by tracing existing vulnerabilities to projects which might directly or indirectly be affected by vulnerabilities inherited from other projects and libraries.

1. Introduction

The Internet has revolutionized our society and impacted the software industry [1], with knowledge and information sharing becoming a central part of software development, facilitating the globalization of the software industry [1]. This change in the information flow removes traditional project boundaries and promotes a free flow of information, resources and knowledge across projects. Globalization in the software industry [2] can have several facets ranging from out- and crowd-sourcing parts of a development process, the wide spread use of collaborative environments facilitating resource sharing, to the introduction of completely new software development paradigms such as open source. Open source software publishes source code and other related artifacts on the Internet using specialized code and artifact sharing portals such

* Corresponding author.
E-mail addresses: s_alqaht@encs.concordia.ca (S.S. Alqahtani), e_eghan@encs.concordia.ca (E.E. Eghan), juergen.rilling@concordia.ca (J. Rilling).

http://dx.doi.org/10.1016/j.scico.2016.01.005
0167-6423/

as Sourceforge,[1] GitHub,[2] and Maven,[3] allowing these artifacts to be shared and reused globally. This reuse can take on different forms, such as integrating open source projects into existing software ecosystems (e.g., reuse of code libraries) or extending and customizing available projects to meet specific requirements, e.g., creating specialized Linux distributions [3].

Shared knowledge resources not only facilitate reuse and collaboration, they also introduce new challenges to the software engineering community. Knowledge and resources are no longer controlled by a single project or organization and instead are now distributed across multiple projects, organizations or even global software ecosystems. Given this new distributed nature of software systems and their resources, traditional analysis approaches and tools, developed for a project level context, no longer scale to the new global software context. Among the challenges arising from the knowledge sharing is Information Security (IS), which has emerged to become a major threat to the software development community. At its core, IS promotes that one should consider different security concepts (e.g. secure programming, knowledge about software security vulnerabilities and their analysis) in the development process. The importance of IS for the software community is reflected by the fact that it has become an integrated part of current software engineering best practices [4]. Different specialized software security and advisory databases (SSD) (e.g., National Vulnerability Database (NVD) [5]) have been introduced to track software known vulnerabilities and potential solutions to resolve them. These SSD can be seen as a direct response by the software industry to the ever-increasing number of software attacks, which no longer are limited to a particular project or computer but often affect now hundreds of different systems and millions of computers.

Software security repositories, like most other software repositories, can be considered trusted information silos that focus on modeling and managing specialized resources or knowledge. Software security repositories are typically not linked to other knowledge resources, such as source code repositories containing reported instances of these problems or project specific issue trackers. There are several reasons why the software community is still dealing with such information silos: 1) Establishing traceability among repositories is often difficult due to the lack of a common, standardized approach for modelling knowledge across repository boundaries [6]. 2) Reasoning and semantic integration are a challenge, since most repositories lack the required expressiveness (e.g. First Order or higher order logics) in their underlying knowledge modeling approach to support automated reasoning [7]. 3) Using relational modeling schemata, facts and schemata remain machine but not human accessible without additional tool support for extracting schemata information, making knowledge sharing more difficult. 4) While significant progress has been made by the mining software repository research community to identify potential (semantic) links among repositories by introducing handcrafted, specialized (trained) machine and data mining techniques within proprietary datasets, a key challenge remains – the results and information obtained remain still not reusable or shareable for later consumption by either humans or machines [8].

Given the growing importance of IS for the software domain and the challenges the software community faces in integrating software repositories, the paper introduces a Semantic Web enabled modeling approach which addresses this IS knowledge integration challenge. While our modeling approach supports the integration of a broad range of traditional software repositories (e.g., issue trackers, version control systems, Q&A repositories), we focus in this research in particular on how we can capture knowledge from specialized IS repositories (e.g., NVD) and seamless integrate this knowledge with one of the more widely used software build repositories (e.g., MAVEN). For our research, we take advantage of the Semantic Web and its supporting technologies such as ontologies and Linked Data to establish a common, standardized unified representation to integrate knowledge resources across existing repository boundaries. In addition, this formal representation allows us take advantage of Semantic Web reasoning services to infer both explicit and implicit knowledge to enrich the already modeled knowledge.

This unified knowledge model allows us to identify potential impact of vulnerabilities on software systems across project boundaries. For example, by linking the NVD security database to the Maven build repository we were able to show that not only 576 projects in Maven directly include known security vulnerabilities, but also that due to the ripple effect more than 400,000 projects or libraries might be potentially affected.

The key benefits of our research are manifold, including the ability to support:

- Transformation of traditional security and vulnerability information silos into information hubs.
- Matching of different ontologies through their shared concepts and semantic linking to support integration of knowledge.
- Identification of potential rippling impact of vulnerabilities across project boundaries based on project build dependencies.

The remainder of this paper is structured as follows: Section 2 introduces the motivation for our research. Section 3 describes in more detail background relevant to our research. Section 4 describes the approach used. Section 5 presents our experiments and findings. We discuss observations regarding our integration approach, and outline the potential threats to the validity of this approach in Section 6. Section 7 discusses relevant work, followed by Section 8, which discusses future work and concludes the paper.

[1] http://sourceforge.net/about.
[2] https://github.com/about.
[3] http://search.maven.org/.