TrustSECO: a tool for Providing Trust in the Software Ecosystem

Fang Hou[1][0000−0002−8042−3278], Siamak Farshidi[2][0000−0003−3270−4398], and Slinger Jansen[1,3][0000−0003−3752−2868]

[1] Utrecht University, the Netherlands f.hou@uu.nl
[2] University of Amsterdam, the Netherlands
[3] Lappeenranta University of Technology, Finland

Abstract. The software ecosystem is a trust-rich part of the world. Collaboratively, software engineers trust major hubs in the ecosystem, such as package managers, repository services, and programming language ecosystems. However, trust entails the assumption of risks. We conducted a set of interviews, a systematic literature review, and systematic mapping to study the software engineers' software selection protocol, trust in software and software ecosystem, and the latest state-of-the-art approaches in software evaluation. Based on this knowledge, in this paper, we present TrustSECO. It is a distributed infrastructure that collects, stores, and discloses trust facts about information systems. With this data software end-users can collectively determine whether packages (versions) are reliable, contain vulnerabilities, and are trusted by other users.

Keywords: Software Trust · Software Selection · Software Engineering.

1 Introduction

The worldwide Software Ecosystem (SECO) concerns all software producing and consuming individuals and organizations, which collaboratively ensure that our society remains functional. Throughout the software life cycle, software engineers, end-users, and other stakeholders collaboratively place their trust in major hubs in the ecosystem, such as package managers, repository services, and software components. The worldwide SECO should be a trust-rich environment, however, it is under constant threat. According to the Sonatype report [10] in 2021 the world witnessed a 650% increase in software supply chain attacks, it is 742% increase in 2022. In terms of the time to remediate, 51% of participants required more than a week to respond. The card house is going to collapse if we do not act fast.

Trust operates under risk and uncertainty [3], which means that without risk and uncertainty, there would be no need for trust to exist. In software engineering, trust is a critical factor influencing their acceptance and effective utilization, especially in assisting software end-users in mitigating perceptions of risk and uncertainty when adopting any uninformed or new technology [9]. Amoroso et al. [1] believe that although programmers, engineers, and managers have traditionally viewed risk and uncertainty as mainly related to security issues, trust extends beyond security, incorporating techniques such as enhanced testing, reviews, and inspections to reduce errors in the software development and maintenance lifecycle. Measuring software trust involves evaluating these components through existing metrics or frameworks. Jensen et al. [6] argue that trustworthiness in software is defined as the level of confidence that software will meet its goals while remaining free from threats under all circumstances. Efforts to establish trustworthiness in software often involve parameters like security, reliability, and maintainability.

According to Nadi et al. [8], software package selection is often a difficult task, with considerable factors, such as technical factors, human factors, and economic factors influencing this selection. While many assume that the SECO hub can perform some quality check or trust analysis, they rarely do. Some expose some simple trust knowledge, such as the number of downloads or 'stars' of appreciation given by the community, but generally, they do not take the responsibility of providing anything more than basic reporting services ("Report a problem with this package").

We propose TrustSECO, which is a community-managed infrastructure that underpins the SECO with a trust layer. The infrastructure gathers data on trust in particular software packages and projects. With this data software end-users can determine whether packages (versions) are reliable, contain vulnerabilities, and are trusted by other users. The purpose is to use empirical software engineering to secure the worldwide SECO.

2 Related Work

We conducted a systematic literature review [2] to explore the software trust in the worldwide SECO, that is, to determine what signifies a trustworthy system or actor. We conduct a systematic literature review on the concept of trust in the SECO. We acknowledge that trust is something between two actors in the SECO, and we examine what role trust plays in the relationships between end-users and (1) software products, (2) package managers, (3) software-producing organizations, and (4) software engineers. Two major findings emerged from the systematic literature review. To begin, we define trust in the SECO by examining the definition and characteristics of trust. Second, we provide a list of trust factors that can be used to assemble an overview of software trust. Trust is critical in the communication between actors in the worldwide SECO, particularly regarding software selection and evaluation. With this comprehensive overview of trust, software engineering researchers have a new foundation to understand and use trust to create a reliable SECO.

Then we conducted a set of interviews [4] to study the trust factors in open-source and closed software selection from a practitioner's perspective, to gain insight into perceptions of software trust. We conducted semi-structured interviews with 24 software practitioners from different businesses, including 12 experts in open-source software selection and 12 experts in closed software selection. By summarizing we found: (1) software selection is affected by three categories of factors, being technical, organizational, and structural assurance factors; (2) the most important factors are technical factors and organizational factors; and (3) software selection process and trust factors differ depending on the project risk tolerance. In addition, we compared the trust factors derived from the interviews with those found in the literature.

A number of approaches have been proposed to help software engineers better evaluate packages. According to a systematic mapping (SM) on software component assessment approaches [5], trust is based on a large number of trust knowledge, the majority of approaches are evidence-based, relying on a set of trust knowledge and weights. However, the factors covered by these approaches are limited. Some software quality attributes, such as security, reliability, maintenance, safety, and availability, are not quantified. Usually, the values and weights of these unqualified attributes are given by experts or from literature. Moreover, evaluating software components frequently becomes a decision-making problem or a complicated mathematical challenge. Calculations of assessment results are relatively complex and not easily understood.

3 Design of the Tool

TrustSECO is a community-managed infrastructure that underpins the SECO with a trust layer. The infrastructure gathers trust facts in particular software packages and projects. With this data software end-users can determine whether packages (versions) are reliable, contain vulnerabilities, and are trusted by other users. The framework is shown in Figure 1. Trust facts are from several sources, including Libraries.io, GitHub, CVE, StackOverflow, and ClamCV. To simplify software engineers' software evaluation or selection process, we integrate TrustECO to npm in the TrustSECO.js tool. It provides pre and post-installation verifications for software packages with comprehensive trust facts and scores from TrustSECO.

The trust facts, shown in Table 1, are informed by the existing literature, including GitHub (development activity), Libraries.io (dependency risk), Stack Overflow (community trust), Common Vulnerabilities and Exposures (CVE) (security vulnerabilities), and ClamAV (malware Detection). CVE serves as a repository of publicly disclosed vulnerabilities [11]. ClamAV is an open-source software for detecting viruses and malware. The trust knowledge weights are based on the study conducted by Mujahid et al. [7]. This study conducts a quantitative analysis of the selection factors based on a set of 2,527 npm packages, subsequently establishing a ranking based on these factors. The trust facts are from a set of interviews with 24 software engineers to discuss the factors influencing their package selection [4]. Their insights were used to refine and adjust this ranking. Finally, we incorporated this enhanced rank to determine the relative contribution of each factor.

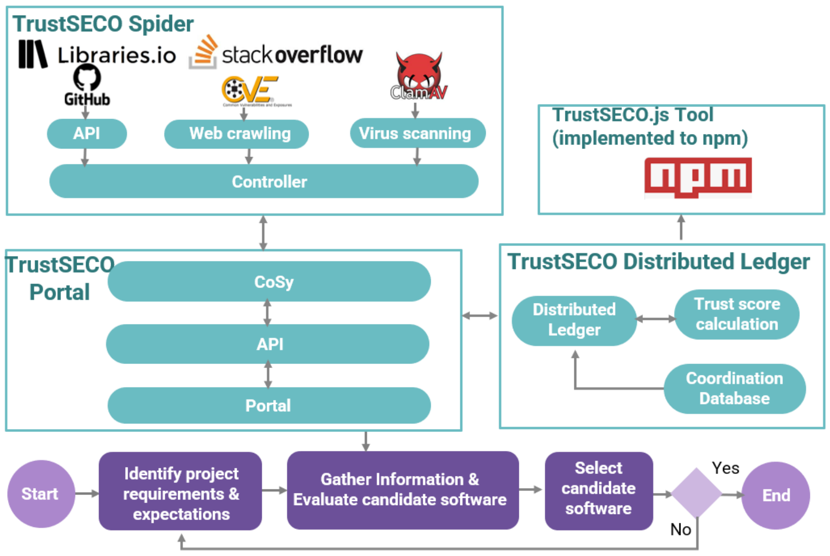

Fig. 1: Overview of TrustSECO framework

According to past interviews, software engineers indicated that numerical formats effectively convey trust results, leading us to present the final trust assessment results in numeric format [4]. The authors also favor simple over complex scoring methods. Consequently, to normalize the varying units of trust knowledge, we employ Min-Max Normalization and calculate a weighted sum as the trust score, ranging from 0 to 100. For a collection of unnormalized trust knowledge factor $F = \{f_1, f_2, \ldots, f_n\}$, where $F_{\min}$ and $F_{\max}$ denote the minimum and maximum values within F, and a set of weights $W = \{w_1, w_2, \ldots, w_n\}$ corresponding to the weight of each trust knowledge factor, the final trust score T is calculated as follows:

$$T = \sum_{i=1}^{n} \left(\frac{f_i - F_{\min}}{F_{\max} - F_{\min}} \cdot w_i \right) \times 100 \tag{1}$$

4 Conclusions

We present TrustSECO, a tool to evaluate software quality. With TrustSECO, software package providers can guarantee that the software that has left software engineers' desks worldwide is the same as the software installed on the end-users' systems and that the software is maintained in a 'healthy' manner. Furthermore, TrustSECO provides users with insight into the trust levels that they can put into the software they use daily. Finally, we hope that TrustSECO can become the plumbing under the trust system that guarantees the software we use to run a modern society.

Table 1: Overview of the Trust knowledge

Trust Knowledge (F)	Sources	Weight (W)
Number of downloads	GitHub	0.1176(+)
Number of downloads of release	GitHub	0.1103(+)
GitHub Stars	GitHub	0.1029(+)
Number of issues without response	GitHub	0.0956(-)
Ratio of open/total issues	GitHub	0.0882(-)
Number of open issues	GitHub	0.0809(-)
Number of dependencies	Libraries.io	0.0735(+)
Number of contributors	GitHub	0.0662(+)
Number of yearly commits	GitHub	0.0588(+)
Number of releases	Libraries.io	0.0515(+)
Release frequency	Libraries.io	0.0441(+)
Number of vulnerabilities	CVE	0.0368(-)
Ratio of detected viruses	ClamAV	0.0221(-)
Number of dependents	Libraries.io	0.0147(+)
StackOverflow popularity	StackOverflow	0.00735(+)
Stargazers	GitHub	Weekly(+)
(+) positive impact, (-) negative impact		
Followings are part of the knowledge but not used for score calculation		
Average issue resolution time	GitHub	
Number of total issues	GitHub	
Repository language	GitHub	
CVE ID	CVE	
CVE score	CVE	
Affected version	CVE	

References

1. Amoroso, E., Taylor, C., Watson, J., Weiss, J.: A process-oriented methodology for assessing and improving software trustworthiness. In: Proceedings of the 2nd ACM Conference on Computer and communications security. pp. 39–50 (1994)
2. anonymous authors: anonymous title. Empirical Software Engineering **28**(1) (2023)
3. Elster, J.: Explaining social behavior: More nuts and bolts for the social sciences. Cambridge University Press (2015)
4. Hou, F., Jansen, F., De Vries, A., Jansen, S.: The role of software trust in selection of open-source and closed software. In: 2023 IEEE/ACM 11th International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems (SESoS). pp. 30–37. IEEE (2023)
5. Hou, F., Jansen, S.: A survey of the state-of-the-art approaches for evaluating trust in software ecosystems. Journal of Software: Evolution and Process p. e2695 (2024)
6. Jensen, C.D., Nielsen, M.B.: Codetrust: Trusting software systems. In: Trust Management XII: 12th IFIP WG 11.11 International Conference, IFIPTM 2018, Toronto, ON, Canada, July 10–13, 2018, Proceedings 12. pp. 58–74. Springer (2018)
7. Mujahid, S., Abdalkareem, R., Shihab, E.: What are the characteristics of highly-selected packages? a case study on the npm ecosystem. Journal of Systems and Software **198**, 111588 (2023)
8. Nadi, S., Sakr, N.: Selecting third-party libraries: the data scientist's perspective. Empirical Software Engineering **28**(1), 15 (2023)
9. Pavlou, P.A.: Institution-based trust in interorganizational exchange relationships: the role of online b2b marketplaces on trust formation. The Journal of Strategic Information Systems **11**(3-4), 215–243 (2002)
10. Sonatype: 2022 state of the software supply chain. The 8th annual state of the software supply chain (2022)

11. Sun, J., Xing, Z., Xu, X., Zhu, L., Lu, Q.: Heterogeneous vulnerability report traceability recovery by vulnerability aspect matching. In: 2022 IEEE International Conference on Software Maintenance and Evolution (ICSME). pp. 175–186. IEEE (2022)