

Poster: Usability, Acceptability, and Efficacy of SBOM Visualization and AI Assistant

Xinyao Ma, Ankith Veldandi, Zitao Zhang, Peter Caven, Ambarish Gurjar, L. Jean Camp
Indiana University Bloomington
(maxiny, aveldan, zhangzit, pcaven, agurjar, ljcamp)@iu.edu

Abstract—The Software Bill of Materials (SBOM) has emerged as a critical tool for mitigating information asymmetry in the software security market. By promoting transparency across the software supply chain, SBOMs provide stakeholders with essential information to support decision-making throughout a product’s lifecycle. This transparency has the potential to systematically improve awareness of vulnerabilities and risks in software development. To make SBOM content more accessible and actionable, multiple visualization tools have been developed. We conducted quantitative and qualitative experiments to evaluate the usability, acceptability, and efficacy of popular SBOM visualization tools. Our findings reveal that while visualizations are helpful, they often fall short in scalability and risk communication, particularly for complex dependency graphs. Motivated by these limitations, we propose BomBot, an LLM-based AI assistant, designed to help developers and stakeholders better understand and interact with software supply chain information.

I. INTRODUCTION

The SBOM is a critical tool in software development, providing a nested inventory of the components that make up a software product [1]. SBOMs are essential for identifying vulnerabilities and ensuring software security [2]. However, the complexity of SBOM data often makes it challenging for developers and stakeholders to interpret and act upon the information effectively.

We aim to develop an AI-powered assistant, BomBot, designed to address these challenges by providing visualization, understanding, vulnerability identification, and suggestions capabilities for SBOMs. BomBot leverages Large Language Models (LLMs) to interact with users, answer questions, and provide actionable insights. The motivation behind BomBot stems from the limitations of current SBOM visualization tools, which, while useful, often fall short in terms of usability, accuracy, and scalability, especially when dealing with large and complex dependency graphs.

In our previous research experiment, we compare the straightforward provision of SBOM data with two popular open-source visualization tools, ItDepends and DeepBits, with the traditional JSON text files. Our results illustrate the need for more effective communication within the current security market. While SBOMs may present an effective and viable option, their current instantiation is not suitable for most consumers. This paper seeks to understand the implications of information conveyance of complex code. Specifically, we focus on how visualizing machine-readable SBOMs can be used to increase transparency and mitigate the vulnerability

of security and privacy. The results indicate that simply visualizing the interdependencies within nested code creates a 144.94% increase in a more accurate selection and a 162.78% increase in risk evaluation of vulnerabilities.

II. METHODOLOGY

The methodology for developing and evaluating BomBot was structured into three main phases: visualization evaluation, chatbot development, and user interaction design. Each phase was designed to address specific challenges in SBOM analysis and vulnerability identification.

A. Visualization Evaluation

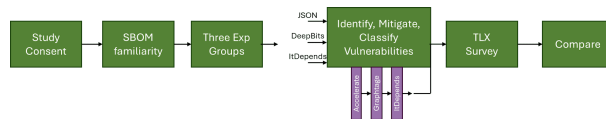


Fig. 1. Experiment design.

Two visualization tools, It-Depends and DeepBits, were chosen for their ability to generate SBOM visualizations from .json files and clear and easy-to-understand structures with nodes connected with lines. The evaluation was conducted using a controlled experiment where participants were asked to interact with SBOM visualizations of varying complexity. The visualizations were assessed based on their usability, acceptability, and accuracy in identifying vulnerabilities and providing mitigation strategies. Usability was measured using the NASA Task Load Index (TLX), which evaluates mental demand, physical demand, temporal demand, performance, effort, and frustration. Acceptability and accuracy were measured through user surveys and task completion rates.

B. Chatbot Development

BomBot was developed as a Retrieval-Augmented Generation (RAG) Bot, leveraging LLMs to answer questions about SBOMs. The chatbot integrates with the Open Source Vulnerability (OSV) database to identify and categorize vulnerabilities by risk level (low, medium, high). The system provides real-time updates on vulnerabilities; however, mitigation information currently requires manual input, as zero-shot training for this task does not consistently yield accurate results.

C. Plan: User Interaction Design

Participants were given tasks to identify vulnerabilities in SBOMs using both visualization tools and the chatbot. The tasks were designed to simulate real-world scenarios where developers need to assess the security of software components.

III. EVALUATION

The evaluation of the SBOM visualization tools was conducted through a combination of quantitative and qualitative measures. The evaluation of BomBot is planned to be conducted as an online AI-assisted vulnerability identify and mitigate human subject experiment, from experts to novices.

A. Visualization Efficacy

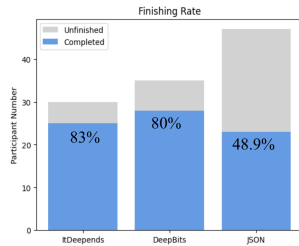


Fig. 2. Acceptability Results: finishing rates for three groups.

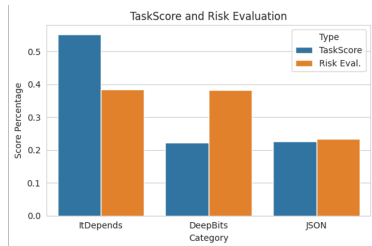


Fig. 3. Accuracy Results: task scores for experiment tasks and risk evaluation of vulnerability packages for three groups.

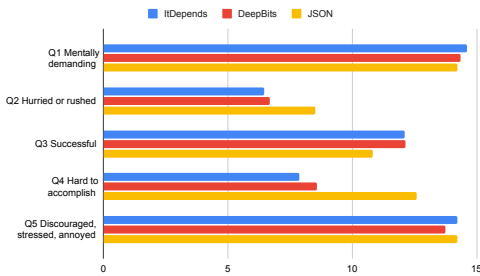


Fig. 4. Usability as Measured by the NASA Task Load Index.

The NASA TLX results indicated that while visualizations were useful, they were also mentally demanding and could be stressful for users. Participants reported high levels of mental effort (Q1) and frustration (Q5) when working on these tasks. Survey results in Fig. 4 showed that 80% to 83% of participants in visualization groups finished all tasks, but only 48.9% in the JSON group finished in Fig. 2. This discrepancy

highlights the utility of SBOM visualizations. Participants also identified vulnerabilities with a high degree of accuracy using the visualizations, see Fig. 3. However, the task was perceived as burdensome, even for small packages, and likely infeasible for graphs with thousands of nodes.

B. Future Plan: BomBot Performance

We expect the BomBot to be able to achieve high accuracy in identifying and categorizing vulnerabilities. Participants could query the BomBot for specific vulnerabilities (e.g., in the Jinja2 package) and receive accurate risk assessments (e.g., high-risk vs. low-risk). In the real-world experiment, we expect participants to report higher satisfaction with the BomBot than with traditional visualization tools. The interactive nature of the BomBot allowed users to ask questions and receive immediate feedback, reducing the cognitive load associated with interpreting complex SBOM data. The bot will then compare the new SBOM with the original, analyze the changes, and provide feedback on how those updates have impacted the software's security.

IV. CONCLUSION

While visualizations were helpful, they were not sufficient for understanding complex SBOM data, especially for large-scale projects. The chatbot, on the other hand, offered a more interactive and scalable solution capable of addressing concerns and answering questions regardless of the complexity of the SBOM. We plan to integrate Microsoft Copilot suggestions with SBOM generation to create a seamless advisory system that provides real-time recommendations during the software development process. Future work will focus on automating risk mitigation information using models like BAAI/bg-m3 and BERT to provide more accurate and consistent mitigation strategies. The team aims to expand BomBot's capabilities to include automated compliance verification, ensuring that software meets regulatory and security standards. Additionally, BomBot will collaborate with GUAC (Graph for Understanding Artifact Composition) for future visualization evaluations and API integration, enhancing the chatbot's ability to process and visualize complex SBOM data.

ACKNOWLEDGMENT

This work was funded by the U.S. Department of Homeland Security under Grant (Award 17STQAC00001-07-00), US Department of Defense (Contract W52P1J2093009), and funding from CTIA. The views and conclusions contained in this poster are those of the authors and should not be interpreted as representing the official policy or opinions expressed or implied by any funder.

REFERENCES

- [1] National Telecommunications and Information Administrator. Framing Software Component Transparency: Establishing a Common Software Bill of Materials (SBOM). 2020.
- [2] National Telecommunications and Information Administration. Software Suppliers Playbook: SBOM Production and Provision. 2021. <https://www.ntia.doc.gov/report/2021/minimum-elements-software-bill-materials-sbom>.



BomBot: A LLM-based AI Agent Assistant for SBOM Visualization, Understanding and Vulnerability Identification

Xinyao Ma, Ankith Veldandi, Zitao Victor Zhang, Peter Caven, Ambarish Gurjar, L. Jean Camp

Luddy School of Informatic, Computing and Engineering, Indiana University

Introduction

- **Software Bill of Materials (SBOM):** a nested inventory for software, a list of ingredients that make up software components.
- **SBOM Visualizations** are designed to translate the data into actionable decisions. We evaluated two visualizations for usability, acceptability, and accuracy
- **BomBot:** LLM-based AI assistant for developers of all levels to leverage SBOMs

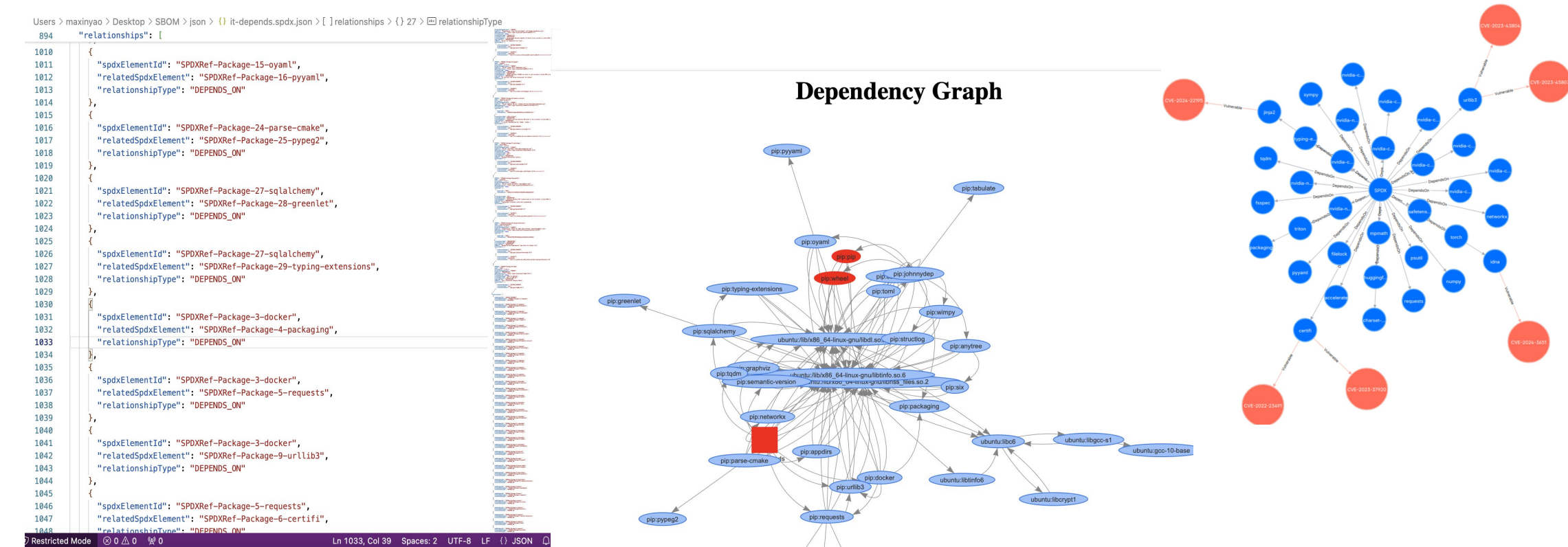


Fig.1 Comparison of a SBOM file in text (JSON), ItDepends, and DeepBits formats

Motivation: SBOM Visualization Efficacy

- **Visualization Tools:**
 - **It-Depends** and **DeepBits** make it feasible to use SBOM data to identify vulnerabilities and the corresponding mitigations.
 - SBOM data maps dependencies, highlights vulnerabilities, and provides real-time CVE updates.

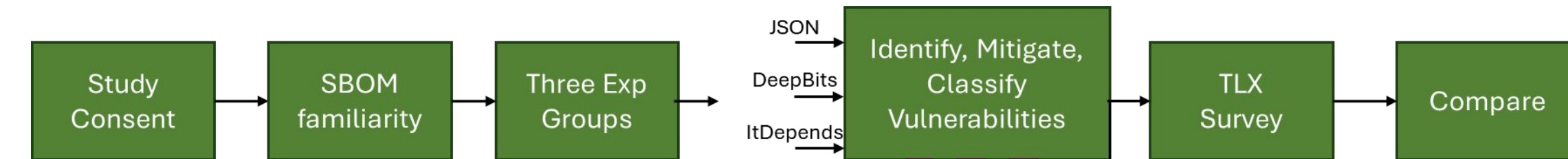


Fig.2 User study experiment design

- **Results**
 - Visualizations were useful in assisting developers in identifying vulnerabilities and mitigations.
 - The task was still perceived as burdensome, even for small packages
 - Likely infeasible when graphs contain thousands of nodes
 - Risk were underestimated even when SBOM data are visualized

Item	Options	n
Gender	Male	51
	Female	16
	Prefer not to answer	45
Age	18-24	34
	25-34	16
	35-44	9
	45-54	5
	55-64	5
	Prefer not to answer	43
SBOM Familiarity	Never heard of it	27
	Have heard of it	34
	Familiar	20
	Knowledgeable	15
	Very knowledgeable	16

Fig.3 Participant Summary

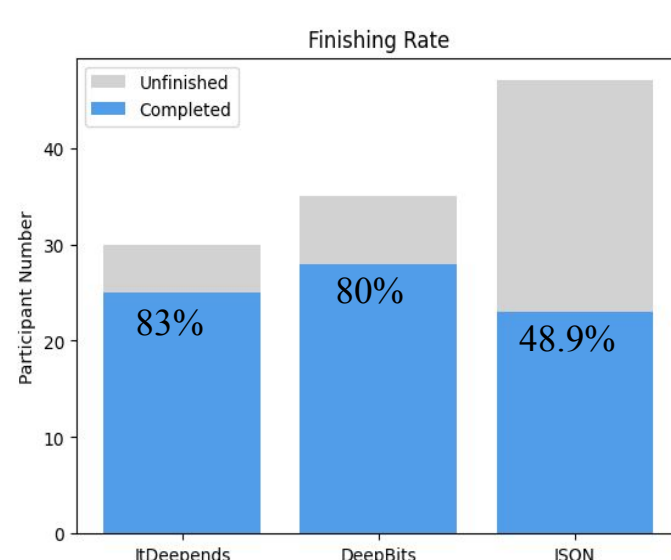


Fig.4 Acceptability Results



Fig.5 Accuracy Results

BomBot Conversation Example

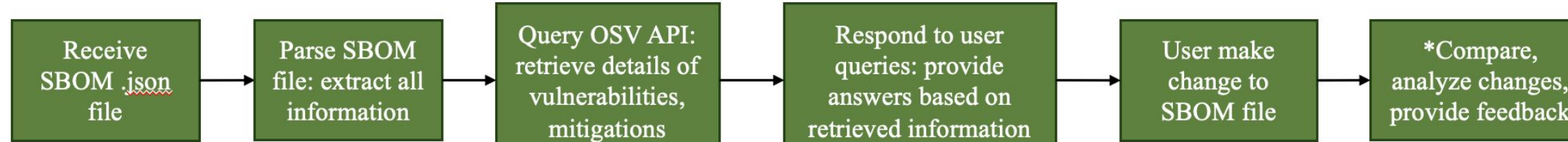
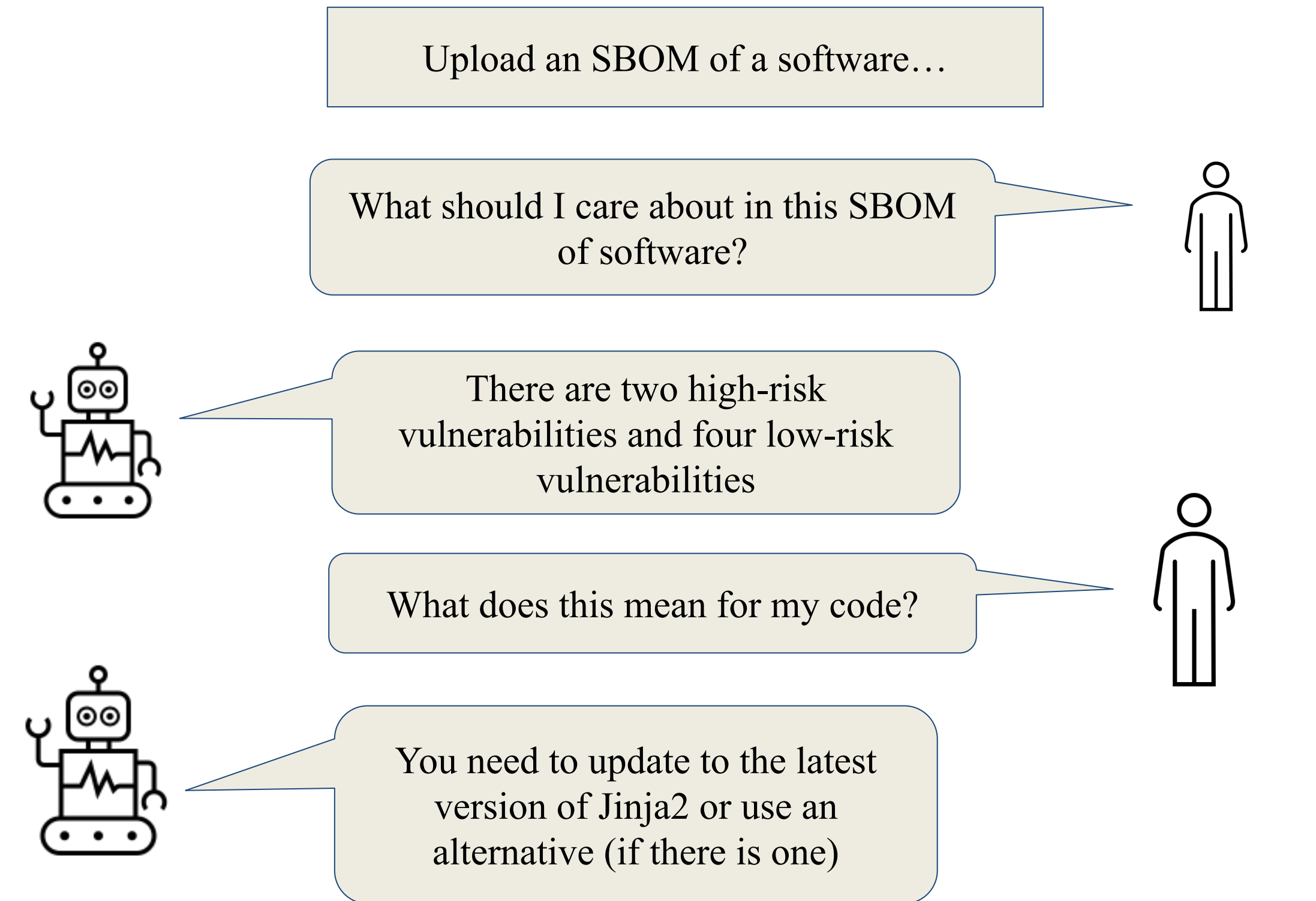


Fig.6 All steps are automated except mitigation information. Mitigation information is verified by researchers. Prompts are automated and invisible to users.

Motivation: SBOM Visualization Usability

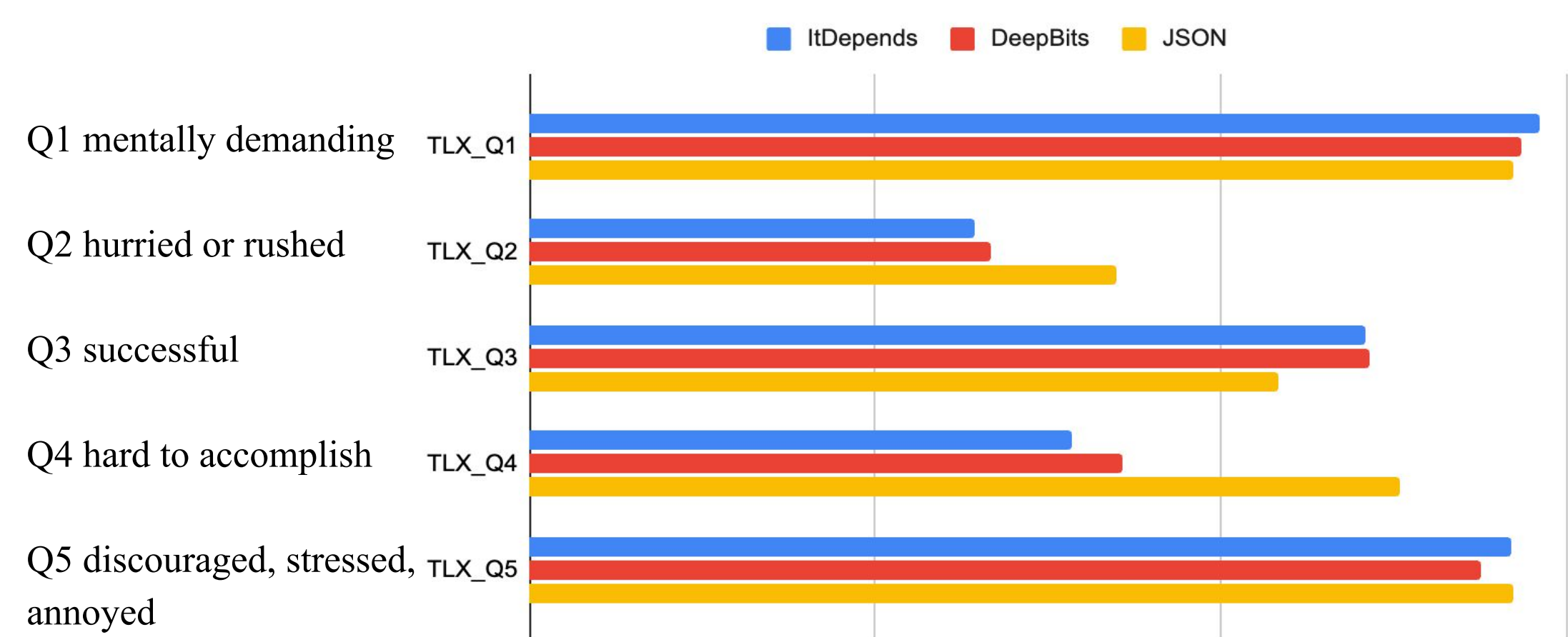


Fig. 7. Usability as Measured by the NASA Task Load Index

Implications

Visualizations are significantly helpful. However, they had limited efficacy even with relatively small graphs.

This motivated us to develop BomBot, a interactive chat system to process accurate SBOM data

Current Functions

A RAG Bot to answer question about SBOMs of known software:

- generate visual graph via .json
- connect with OSV database
- identify known vulnerabilities and risk level

Output for question:

In this SBOM, what are the vulnerabilities in the Jinja2 package? Also, categorise it in between low, medium and high risk level.

```

The Jinja2 package has two notable vulnerabilities:
1. **Vulnerability ID: GHSA-h5cd-r0pp-c9ss**
- **Summary:** Jinja is vulnerable to HTML attribute injection when user input is passed as keys to the 'mlattr' filter.
- **Details:** The 'mlattr' filter accepts keys containing spaces, which are not allowed in XML/HTML attributes. If an application accepts keys as user input and renders them in pages visible to other users, an attacker could inject other attributes and perform cross-site scripting (XSS). This is not a common or intended use case for the 'mlattr' filter, and applications should verify keys regardless of this fix.
- **Severity:** The CVSS score is CVSS:3.1/AV:N/AC:L/PR:N/UI:R/SU:C/L/I:N/AW, indicating a low to medium risk. The score suggests that the attack can be executed over the network, requires low attack complexity, no privileges, and user interaction, with a low impact on confidentiality and integrity.
2. **Vulnerability ID: GHSA-h75v-3v3j-5mfj**
- **Summary:** Similar to the first vulnerability, Jinja is vulnerable to HTML attribute injection when user input is passed as keys to the 'mlattr' filter.
- **Details:** The 'mlattr' filter accepts keys containing non-attribute characters like spaces, '/', '>', or '=', which can be interpreted as starting separate attributes. This vulnerability allows attacker to inject other attributes and perform XSS. Accepting keys as user input is now considered an unintended use case, and code doing so without validation should be flagged as insecure.
- **Severity:** The CVSS score is the same as the first vulnerability, indicating a low to medium risk.

## Vulnerability Rating Explanation
The CVSS (Common Vulnerability Scoring System) score provides a way to capture the principal characteristics of a vulnerability and produce a numerical score reflecting its severity. Here's a step-by-step explanation of the score components:
- **AV:N (Attack Vector: Network):** The vulnerability can be exploited over a network.
- **AC:L (Attack Complexity: Low):** The attack does not require special conditions.
- **PR:N (Privileges Required: None):** No privileges are required to exploit the vulnerability.
- **UI:R (User Interaction: Required):** Exploitation requires user interaction.
- **SU:U (Scope: Unchanged):** The exploitation of the vulnerability does not affect resources beyond its scope.
- **C:IL (Confidentiality Impact: Low):** Limited impact on confidentiality.
- **I:IL (Integrity Impact: Low):** Limited impact on integrity.
- **A:N (Availability Impact: None):** No impact on availability.

## Risk Categorization
Both vulnerabilities can be categorized as **medium risk** due to their potential to allow XSS attacks, which can compromise user data and application integrity, especially in applications that do not validate user input properly.

```

Conclusions & Limitations

Conclusions

- Visualizations help but are not adequate. Visualizations can differ significantly in efficacy even when they are equally usable. One risk of both SBOM and the associated visualizations is that people may be unduly confident that they have a full understanding of their attack space, when in fact they are underestimating the risks.
- A chatbot offers an interactive method that can address concerns and answer questions regardless of scale. A chatbot can rank and integrate information.

Limitations

- There are multiple visualizations for the SBOM. Our work tested only two options. Innovative or next iteration of visualization will improve.
- Text requires more attention and may interrupt programming while a visualization can provide continuous if passive feedback.
- Our sample was recruited primarily from University computer science students and using Prolific for people with computing backgrounds. There were fewer than 20 industry participants, these were not significantly different. Repeating the experiment with a private sector software engineering team may yield different results.

Future Work

- Allow users to upload new, updated SBOMs after making changes and bot will compare the new with the original, analyze the changes, and provide feedback.
- Evaluate expansions to improve acceptability, e.g. automate compliance verification
- Collaborate with GUAC team from Google for future visualization evaluation and API for chatbot.
- We are just beginning to understand how to merge risk communication with SBOMs.

Acknowledgment: This work was funded by the U.S. Department of Homeland Security under Grant (Award # 17STQAC00001-07-00), US Department of Defense (Contract W52P1J2093009) and funding from CTIA. The views and conclusions contained in this poster are those of the authors and should not be interpreted as representing the official policy or opinions expressed or implied by any funder.