Accepted at FIE26

WIP: Educating Students on Kubernetes-Related Policy Violations: Can We Use Static Application Security Testing?

Akond Rahman* Fan Wu† Hossain Shahriar‡
* Auburn University, AL, USA
† Tuskegee University, AL, USA
‡ University of West Florida, FL, USA
Email: *akond@auburn.edu †fwu@tuskegee.edu ‡hshahriar@uwf.edu

***Abstract*—This research work-in-progress (WIP) paper describes experiences from using a static application security testing (SAST) tool for educating students on violations of security-related policies as these violations in software systems incur serious consequences, such as monetary losses for organizations. Software systems that use Kubernetes, a popular container orchestration tool, are no different. Hence, it is important for educators to teach students about security-related policy violations. *The goal of this paper is to help educators teach policy-related violations to upper-division students enrolled in computer science courses by conducting an exploratory study with a static application security testing tool.* We investigate if static application security testing can aid students in learning security-related policy violations for Kubernetes. We use a SAST tool called Kubescape for an exercise that is conducted at a university. Based on the feedback collected from 66 students, we find usage of the SAST tool to be perceived as useful by students to learn about violations of security-related policies in Kubernetes configuration files. Based on our findings, we recommend educators to integrate exercises to help students learn about Kubernetes-related policy violations. We also advocate educators to investigate the usage of SAST tools for enhancing students' knowledge on foundational concepts related to security policies.**

***Index Terms*—kubernetes, security policy, testing**

I. Introduction

A large language model (LLM) is an artificial intelligence (AI) technique that is trained on massive text corpora, which can generate content in response to queries [3], [15]. According to one industry survey, by 2030 LLMs will be used by 45 million professionals who develop software across the globe [5]. With the help of LLMs, professionals complete software engineering tasks in 55% less time than that of not using LLMs [6]. LLMs have shown promise in generating software artifacts, such as source code and test data [4], [7].

Similar to software artifacts, LLMs have shown promise for generating configuration files that are used to automatically manage Kubernetes-based infrastructure [18]. Kubernetes is the most popular tool to implement the practice of container orchestration [27]. Container orchestration is the practice of pragmatically managing container at scale [17]. Since, the use of containers for software deployment have become widespread in software industry, managing numerous containers manually can become error-prone and time consuming [27]. Kubernetes alleviates these shortcomings by enabling practitioners to manage multiple containers at scale. For example, Uber uses more than 500,000 containers to deploy their software services. Usage of Kubernetes helps them in reducing complexity related to container management and rapidly deploy their software services to end users [14]. As another example, OpenAI reported Kubernetes usage to facilitate a reduction in deployment time from "a couple of months" to "two or three days" [10].

Despite reported benefits, configuration files used for Kubernetes can contain security vulnerabilities, such as configurations that violate Kubernetes-related security policies [17], [20]. These vulnerabilities can be leveraged by adversaries to conduct security attacks, such as the infamous Tesla attack [8]. Security vulnerabilities are also prevalent in LLM-generated configuration files. Zhang et al. [26] found 208 instances of security policy violations in 98 configuration files that are generated ChatGPT. Figure 1 shows an example of a configuration that violates a Kubernetes-related security policy called 'ensure memory limits are set'. This security policy is provided by the US National Security Agency (NSA), which recommends to provide memory consumption limits for the containers that will be managed with Kubernetes [1]. Without setting memory limits for containers, the Kubernetes deployment becomes susceptible to denial of service attacks [17]. The code snippet is generated by ChatGPT and obtained from the DevGPT dataset [25].

The above-mentioned showcases the importance to educate students and practitioners on violations of security policies so that their Kubernetes-based deployment infrastructure are not susceptible to attacks. The use of LLMs in the software industry further highlights the importance of educating students on detecting security policy violations in LLM-generated configuration files for Kubernetes.

[1]https://www.armosec.io/blog/nsa-cisa-kubernetes-hardening-guide/

```
kind: Deployment
metadata:
  name: influxdb
spec:
...
  spec:
    containers:
    - name: influxdb
      image: influxdb:1.8
      ports:
      - containerPort: 8086
      volumeMounts:
      - name: influxdb-conf
        mountPath: /etc/influxdb/influxdb.conf
        subPath: influxdb.conf
        readOnly: true
```

Fig. 1: An example of a code snippet generated by ChatGPT that includes a violation of a security policy called 'Ensure memory limits are set'. This security policy is recommended by the NSA framework.

The goal of this paper is to help educators teach policy-related violations to upper-division students enrolled in computer science courses by conducting an exploratory study with a static application security testing tool.

Since use of static application security testing (SAST) tools is useful to enhance security for the software [12] and also educate students on secure software development [16], students can benefit from a hands-on exercise to learn about violations of security policy violations using a SAST tool. To that end, we develop and integrate an exercise that uses a SAST tool to educate students on security policy violations in LLM-generated configuration files. In this paper, we describe our experiences in constructing and disseminating an exercise where students apply a SAST tool called Kubescape [11] to detect violations of security-related policies in configuration files generated by ChatGPT. While there are multiple SAST tools, such as Trivy and SLI-KUBE, we select this tool because of it superior coverage with respect to detecting multiple categories of security-related policy violations [20].

Contributions: We list our contributions as follows:

- An evaluation of using a SAST tool to educate students on security policy violations for configuration files; and
- An evaluation of an exercise that uses LLM-generated configuration files to educate students on the code quality of LLM-generated software artifacts.

II. Related Work

Our paper is related to prior research on security of Kubernetes-based systems. Shamim et al. [8] systematized Kubernetes security practices by analyzing 104 internet artifacts. Blaise et al. [2] used topologies of configuration files to quantify which parts of a Kubernetes-based deployment are more likely to have security consequences. A comparative study by Kamieniarz et al. [9] analyzed the security postures of four Kubernetes clusters in distinct configurations, including two on-premises clusters and two managed services from public cloud providers. Prior work from Rahman et al. [17] and Shamim et al. [19], [20] focused on quantifying security weaknesses in OSS configurations files. The publication that is closest in spirit to our paper is research conducted by Zhang et al. [26]. Zhang et al. [26] used the DevGPT dataset to quantify maintenance and security-related bugs in configuration files that are used for Kubernetes-based deployments.

The above-mentioned discussion shows extensive research on Kubernetes reliability, testing, and security. However, there is a lack of publications that have investigated using security tools to educate students on Kubernetes-related policy violations. We address this research gap in our paper. In particular, we address: (i) a description of an exercise related to using a SAST tool with Kubernetes configuration files generated by ChatGPT; and (ii) quantify students' self-reported knowledge on Kubernetes-related security policies and LLM-generated configuration files after completing the exercise.

The novelty of our paper is also based on the lack of prior work that have investigated on how to teach students about security-related policy violations. In existing education-related research, we have found researchers to investigate effectiveness of cybersecurity programs [1], effectiveness of keystroke logging [24], application of virtual reality for cybersecurity [23], massively open online courses [22], and effectiveness of pedagogical frameworks, such as authentic learning [21]. However, there is a lack of research on how to educate students on security-related policy violations with SAST tools. To the best of our knowledge, no prior work has investigated the use of SAST tools specifically to educate students on Kubernetes security policy violations in LLM-generated configuration files.

III. Methodology

A. Background

1) Configuration Files in Kubernetes: Kubernetes is the most popular tool to implement container orchestration [27]. Kubernetes allows its users to specify properties of a software deployment infrastructure in the forms of YAML files. For example, Listing 1 hows a deployment infrastructure called 'simple-deployment' for a software application called 'simple-app'. The software is deployed using containers that use an image called 'nginx:1.25'. With three 'replicas', the deployment infrastructure ensures it will use three 'pods' each with multiple containers. A pod is the most fundamental deployment unit in Kubernetes.

2) Policies on Kubernetes: Multiple organizations provide guidelines on how to provision them. One of the agencies is the U.S. National Security Agency (NSA), which has introduced the 'Kubernetes Hardening Guide' [13]. This document provides multiple policies on how to securely configure a Kubernetes-based deployment. We select this document to demonstrate security policies.

```yaml
kind: Deployment
metadata:
  name: simple-deployment
  labels:
    app: simple-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: simple-app
  template:
    metadata:
      labels:
        app: simple-app
    spec:
      containers:
        - name: my-container
          image: nginx:1.25
          ports:
            - containerPort: 80
```

Listing 1: An example of a configuration file that specifies a Kubernetes-based deployment.

3) KubeScape: KubeScape is a popular SAST for Kubernetes-based deployments. The tool is developed by ARMO that supports risk analysis and detecting violations of security compliance inside a Kubernetes deployment [11]. While there are multiple SAST tools, we select this tool because of its superior coverage with respect to detecting multiple categories of security-related policy violations [20].

B. Application of KubeScape for Education

The first author of the paper performs two activities:

1) Presentation of KubeScape in Class: The first author of the paper, who is an instructor of a course titled 'Secure Software Process' dedicate one lecture to demonstrate KubeScape. First, at the beginning of the lecture, the first author provides necessary context on container orchestration and Kubernetes. Then they showcase the NSA guideline that lists security-related policies. Second, the first author showcases KubeScape by visiting the webpage for the tool and by executing the tool from command line. As part of this step, the first author explains how the tool works and how it applies static code analysis to find security-related policy violations in configuration files. Third, the first author executes KubeScape with a set of configuration files used in Kubernetes-based deployment. These configuration files are downloaded from prior work that conducted Kubernetes-related research [17].

2) Application of KubeScape by Students: Here, the first author asks their students to use the tool themselves as part of a class exercise. Each student is asked to execute KubeScape with a set of configuration files generated by ChatGPT. This set of files are available as part of the DevGPT dataset [25], which contains a set of software artifacts that are generated by ChatGPT. After executing the tool, each student is asked to report the count of violations for each policy category.

C. Online Survey

Once the exercise is complete, the first author asks the student to participate in an online survey developed with Qualtrics [2]. In the survey each student is asked:

- "How would you rate your knowledge in NSA-provided security policies for Kubernetes before the exercise? " This is a Likert item question with the following options: 'Extremely Knowledgeable', 'Knowledgeable', 'Moderately Knowledgeable', 'Little Knowledge', 'No knowledge'
- "How would you rate your knowledge about the code quality for LLM-generated configuration files before the exercise?" This is a Likert item question with the following options: 'Extremely Knowledgeable', 'Knowledgeable', 'Moderately Knowledgeable', 'Little Knowledge', 'No knowledge'
- "How would you rate your knowledge in NSA-provided security policies for Kubernetes after the exercise? " This is a Likert item question with the following options: 'Extremely Knowledgeable', 'Knowledgeable', 'Moderately Knowledgeable', 'Little Knowledge', 'No knowledge'
- "How would you rate your knowledge about the code quality for LLM-generated configuration files after the exercise?" This is a Likert item question with the following options: 'Extremely Knowledgeable', 'Knowledgeable', 'Moderately Knowledgeable', 'Little Knowledge', 'No knowledge'
- "To what extent the following was useful for you to learn about NSA-provided security policies for Kubernetes? " This is a Likert item question with the following options: 'Extremely useful', 'Useful', 'Moderately useful', 'Little useful', 'Not at all useful'. We ask this question for two items: 'In-class discussion' and 'Using the tool myself'
- "To what extent the following was useful for you to learn about the code quality of LLM-generated configuration files? " This is a Likert item question with the following options: 'Extremely useful', 'Useful', 'Moderately useful', 'Little useful', 'Not at all useful'. We ask this question for two items: 'In-class discussion' and 'Using the tool myself'

[2]https://www.qualtrics.com/

IV. Results

We collect feedback from 66 students who are enrolled in a course titled 'Secure Software Process' at University of Z. Of these students, five are graduate students and the remaining students are all senior year undergraduate students. Their self-reported knowledge in security-related policies before and after conducting the exercise is shown in Figure 2. For example, the percentage of students who report 'knowledgeable' for NSA-provided security policies increased from 15.1% to 65.2% after the exercise.

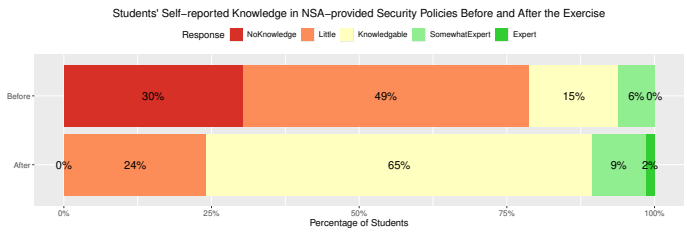


Fig. 2: Students' self-reported knowledge on NSA-provided security policies before and after the exercise.

The students' self-reported knowledge in LLM-generated configuration files before and after conducting the exercise is shown in Figure 3. For example, the percentage of students who report 'knowledgeable' for LLM-generated configuration files increased from 51.5% to 63.6% after the exercise.

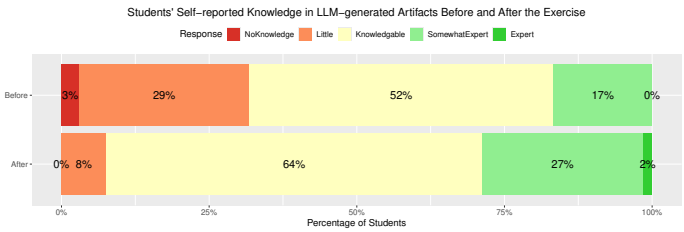


Fig. 3: Students' self-reported knowledge on LLM-generated artifacts before and after the exercise.

We also provide the perceptions of the students on the in-class discussion and the exercise in regards to usefulness in Figure 4. Figure 4a and 4b respectively, provides perceptions of students for learning NSA-provided security policies and code quality of LLM-generated configuration files. From Figure 4a we observe the proportion of students who find tool usage 'Extremely useful' for learning NSA-provided security policies is 11%, whereas for in-class discussion it is 23%. From Figure 4b we observe the proportion of students who find tool usage 'Useful' for learning code quality of LLM-generated configuration files is 76%, whereas for in-class discussion it is 61%. We find in-class discussion to be more useful for NSA-provided security policies than that of code quality of LLM-generated configuration files. We find tool usage to be more useful for the code quality of LLM-generated configuration files than that of NSA-provided security policies.

V. Discussion and Conclusion

A. Discussion

We discuss the implications for educators as follows:

- Classroom discussion helps students to learn about security policies. Therefore, educators who teach security policies should discuss in details about policies, their origins, and other contextual details.
- Classroom discussion on security policies can be complemented by tool usage as usage of tools are perceived to be useful for students.
- Usage of exercises help students to learn about security policies. We urge educators who have not introduced security polices in their curriculum to incorporate topics related to security policies in the forms of exercises.
- Education researchers should build upon our work to investigate if the exercise can actually aid in increasing learning outcomes and concepts on security-related policies. Researchers should also investigate the effectiveness of tools and in-class discussion for teaching students on security policies on other software development topics.

B. Limitations

While our empirical study is grounded with survey data, there are limitations that provide threats to the validity of the paper. *First*, we do not use a control group and a treatment group, which may limit the impact of the conducted activity. *Second*, instead of outcome-based measures, we use self-reported perception data, which may not be reflective of students' actual learning ability on this topic. This is an example of a 'mono-method' bias, where only one intervention is investigated by the researchers. *Third*, there is a lack of direct assessment of student performance. *Fourth*, our results are limited to the use of one tool called Kubescape. Usage of other tools, such as Trivy and SLI-KUBE may yield results that are different than that of reported in the paper. *Finally*, our empirical study is susceptible to generalizability as the data is collected from one course conducted at one university.

C. Conclusion

In this WIP paper, we have provided our experiences on disseminating an exercise in a classroom to teach about security policies. As part of the exercise, we discussed the context of Kubernetes-related security policies and provided the students an exercise on using a SAST tool, to identify security policy violations in configuration files generated by ChatGPT. From our survey results, we observe students to find classroom discussion and tool usage to be useful for learning security policies and their violations in configuration files. While these results show promise, we advocate researchers to build upon our work to investigate if tools and exercise can help in increasing knowledge.

Acknowledgments

We thank the PASER group at Auburn University for their valuable feedback. This research was partially funded by the U.S. National Science Foundation (NSF) Award # 2310179 and Award # 2312321.

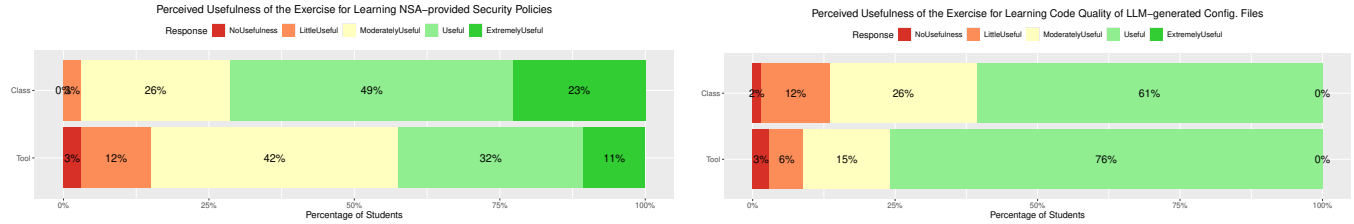
References

[1] S. K. Beach, "Usable cybersecurity: Human factors in cybersecurity education curricula," *Nat. Cybersecur. Inst. J*, vol. 1, no. 1, pp. 5–15, 2014.

[2] A. Blaise and F. Rebecchi, "Stay at the helm: secure kubernetes deployments via graph generation and attack reconstruction," in *2022 IEEE 15th International Conference on Cloud Computing (CLOUD)*, 2022, pp. 59–69.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.

[4] X. Du, M. Liu, K. Wang, H. Wang, J. Liu, Y. Chen, J. Feng, C. Sha, X. Peng, and Y. Lou, "Evaluating large language models in class-level code generation," in *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, ser. ICSE '24. New York, NY, USA: Association for Computing Machinery, 2024.



a b

Fig. 4: Students' perceived usefulness on in-class discussion and tool usage in regards to learning about NSA-provided security policies and code quality of LLM-generated artifacts. Figures 4a and 4b respectively, presents the students perceptions on learning about NSA-provided security policies and code quality of LLM-generated configuration files.

[5] GitHub, "The economic impact of the ai-powered developer lifecycle and lessons from github copilot," https://github.blog/news-insights/research/the-economic-impact-of-the-ai-powered-developer-lifecycle-and-lessons-from-github-copilot/, 2026, [Online; accessed 23-March-2026].

[6] ——, "Research: quantifying github copilot's impact on developer productivity and happiness," https://github.blog/news-insights/research/research-quantifying-github-copilots-impact-on-developer-productivity-and-happiness/, 2026, [Online; accessed 26-March-2026].

[7] M. M. Hassan, J. Salvador, S. K. K. Santu, and A. Rahman, "State reconciliation defects in infrastructure as code," *Proc. ACM Softw. Eng.*, vol. 1, no. FSE, jul 2024. [Online]. Available: https://doi.org/10.1145/3660790

[8] M. S. Islam Shamim, F. Ahamed Bhuiyan, and A. Rahman, "Xi commandments of kubernetes security: A systematization of knowledge related to kubernetes security practices," in *2020 IEEE Secure Development (SecDev)*, 2020, pp. 58–64.

[9] K. Kamieniarz and W. Mazurczyk, "A comparative study on the security of kubernetes deployments," in *2024 International Wireless Communications and Mobile Computing (IWCMC)*, 2024, pp. 0718–0723.

[10] Kubernetes, "Kubernetes user case studies," Online, 2025, accessed: 2025-12-30. [Online]. Available: https://kubernetes.io/case-studies/

[11] kubescape, "Kubescape," https://kubescape.io/, 2026, [Online; accessed 21-March-2026].

[12] G. McGraw, *Software security: building security in*. Addison-Wesley Professional, 2006, vol. 1.

[13] NSA, "Kubernetes hardening guide," https://media.defense.gov/2022/Aug/29/2003066362/-1/-1/0/CTR_KUBERNETES_HARDENING_GUIDANCE_1.2_20220829.PDF, 2022, [Online; accessed 23-March-2026].

[14] Perfology, "Kubernetes at Uber Scale : Scaling Thousands of Services and Millions of Containers Daily," https://www.youtube.com/watch?v=H_4j_M_J1Bc, 2024, [Online; accessed 05-March-2026].

[15] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI*, 2019, accessed: 2024-11-15.

[16] A. Rahman, H. Shahriar, and D. B. Bose, "How do students feel about automated security static analysis exercises?" in *2021 IEEE Frontiers in Education Conference (FIE)*, 2021, pp. 1–4.

[17] A. Rahman, S. I. Shamim, D. B. Bose, and R. Pandita, "Security misconfigurations in open source kubernetes manifests: An empirical study," *ACM Trans. Softw. Eng. Methodol.*, vol. 32, no. 4, May 2023. [Online]. Available: https://doi.org/10.1145/3579639

[18] P. Sahoo, S. Pujar, G. Nalawade, R. Genhardt, L. Mandel, and L. Buratti, "Ansible lightspeed: A code generation service for it automation," in *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, ser. ASE '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 2148–2158.

[19] S. I. Shamim, H. Hu, and A. Rahman, "Dynamic application security testing for kubernetes deployment: An experience report from industry," in *Proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering*, 2025, pp. 514–519.

[20] ——, "On prescription or off prescription? an empirical study of community-prescribed security configurations for kubernetes," in *2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE)*. IEEE Computer Society, 2025, pp. 707–707.

[21] S. I. Shamim, F. Wu, H. Shahriar, A. Skjellum, and A. Rahman, " Authentic Learning Exercise for Kubernetes Misconfigurations: An Experience Report of Student Perceptions ," in *2025 IEEE/ACM 37th International Conference on Software Engineering Education and Training (CSEET)*. Los Alamitos, CA, USA: IEEE Computer Society, May 2025, pp. 292–302. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CSEET66350.2025.00037

[22] C. Theisen, L. Williams, K. Oliver, and E. Murphy-Hill, "Software security education at scale," in *Proceedings of the 38th International Conference on Software Engineering Companion*, 2016, pp. 346–355.

[23] S. V. Veneruso, L. S. Ferro, A. Marrella, M. Mecella, and T. Catarci, "Cybervr: An interactive learning experience in virtual reality for cybersecurity related issues," in *Proceedings of the International Conference on Advanced Visual Interfaces*, ser. AVI '20. New York, NY, USA: Association for Computing Machinery, 2020.

[24] C. Wood and R. Raj, "Keyloggers in cybersecurity education." in *Security and Management*. Citeseer, 2010, pp. 293–299.

[25] T. Xiao, C. Treude, H. Hata, and K. Matsumoto, "Devgpt: Studying developer-chatgpt conversations," in *Proceedings of the 21st International Conference on Mining Software Repositories*, ser. MSR '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 227–230. [Online]. Available: https://doi.org/10.1145/3643991.3648400

[26] Y. Zhang, R. Meredith, W. Reeves, J. Coriolano, M. A. Babar, and A. Rahman, "Does generative ai generate smells related to container orchestration?: An exploratory study with kubernetes manifests," in *Proceedings of the 21st International Conference on Mining Software Repositories*, ser. MSR '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 192–196.

[27] Y. Zhang, U. Paul, M. dx27;Amorim, and A. Rahman, " Configuration Defects in Kubernetes ," *IEEE Transactions on Software Engineering*, no. 01, pp. 1–21, Feb. 5555. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/TSE.2026.3659785