IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 28 March 2022, accepted 13 June 2022, date of publication 21 June 2022, date of current version 1 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3185069

Automated Risk Management Based Software Security Vulnerabilities Management

RAGHAVENDRA RAO ALTHAR[1,2], DEBABRATA SAMANTA[3], (Member, IEEE), MANJIT KAUR[2], (Senior Member, IEEE), DILBAG SINGH[2], (Senior Member, IEEE), AND HEUNG-NO LEE[2], (Senior Member, IEEE)

[1]Data Science Department, CHRIST University, Bangalore, Karnataka 560029, India
[2]QMS, First American India Private Ltd., Bangalore, Karnataka 560038, India
[3]Department of Computer Science, CHRIST University, Bangalore, Karnataka 560029, India
[4]School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology, Gwangju 61005, South Korea

Corresponding author: Heung-No Lee (heungno@gist.ac.kr)

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-01842, Artificial Intelligence Graduate School Program (GIST)) and This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2021-0-01835) supervised by the IITP(Institute of Information & Communications Technology Planning & Evaluation).

ABSTRACT An automated risk assessment approach is explored in this work. The focus is to optimize the conventional threat modeling approach to explore software system vulnerabilities. Data produced in the software development processes are better leveraged using Machine Learning approaches. A large amount of industry knowledge around security vulnerabilities can be leveraged to enhance current threat modeling approaches. Work done here is in the ecosystem of software development processes that use Agile methodology. Insurance business domain data are explored as a target for this study. The focus is to enhance the traditional threat modeling approach with a better quantitative approach and reduce the biases introduced by the people who are part of software development processes. This effort will help bridge multiple data sources prevalent across the software development ecosystem. Bringing these various data sources together will assist in understanding patterns associated with security aspects of the software systems. This perspective further helps to understand and devise better controls. Approaches explored so far have considered individual areas of software development and their influence on improving security. There is a need to build an integrated approach for a total security solution for the software systems. A wide variety of machine learning approaches and ensemble approaches will be explored. The insurance business domain is considered for the research here. CWE (Common Weaknesses Enumeration) mapping from industry knowledge are leveraged to validate the security needs from the industry perspective. This combination of industry and company data will help get a holistic picture of the software system's security. Combining the industry and company data helps lay down the path for an integrated security management system in software development. The risk management framework with the quantitative threat modeling process is the work's uniqueness. This work contributes toward making the software systems secure and robust with time.

INDEX TERMS Quantitative threat modeling, software security, machine learning, quantitative risk assessment, integrated security management system.

I. INTRODUCTION

Threat modeling is one of the prominent parts of software development processes. A large part of the exercise includes expert judgment in practice. There is a need to make this exercise as quantitative as possible. In this paper, constructs of threat modeling are studied to build a quantitative threat modeling approach with less dependency on the experts. The paper is organized to explore some of the related work in the field. The study's objective is to enable a more extensive system of smart security in software development [1], [2]. This smart security system is discussed in detail in the earlier work by the author [3], [4]. Threat modeling and its constructs are described. Exploration of risk assessment

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Ali Babar.

methodology for threat modeling is conducted. Data collection and modeling for threat prediction are covered. Paper wraps up with the recommendations and future work. The focus of the work is automating the security vulnerability risk assessment approach and threat modeling approach with the machine learning approach. Both exercises are optimally combined for better outcomes. Machine learning classification approaches are leveraged to get visibility into possible security vulnerabilities.

A. MOTIVATION

This work is motivated by the lack of focus on software vulnerabilities threat modeling. Though this exercise is conducted, it is restricted due to the manual intervention involved by experts and the time involved. Multiple efforts focus on the security vulnerability discovery but are happening in silos. This state motivates to bring together the efforts into a common framework. With fast-paced progress in the security threat and its impact, it is essential to develop these systems.

B. OBJECTIVE

This work intends to quantify the security threats and help focus as needed. Connecting the knowledge prevalent in the industry with the needs of the software development industry is the focus area. Utilizing the historical knowledge of the organization for better visibility into future operations is given prominence in the study. The flexible threat modeling approach targets the critical security areas as per the area prioritized under study. Getting the confidence of the software development stakeholders with predictable and secure software systems.

C. CONTRIBUTION

Paper contributes an approach to making threat modeling a data-based quantitative process, by reducing the manual intervention of the experts. This approach reduced the dependency on the security experts. With reduced dependency on experts and human intervention, this approach can be extensively used when needed. The proposed approach will help build a knowledge system that will get better over time by including knowledge from across the industry and within the company. This proposed system helps bridge the gap between security experts, software development teams, and software system users. This work is part of a comprehensive Software Security Management system envisioned by the authors.

Paper also contributes to the Information Security domain by helping reconcile the data available across industry and companies for the benefit of software development teams. The ideas presented in this paper are noble and essential in ensuring a common approach to threat modeling in an organizational setting. An integrated model for detecting security threats in an organizational setting will help the software development teams explore security flaws effectively. This approach would be of immense use as it standardizes threat detection in software system development. It will serve as a knowledge management tool for software development companies. The primary focus of our work is on applying data analytics in threat modeling, and risk assessment approaches and proposing an integrated approach.

The paper starts with exploring the literature for work done on threat modeling for software development and machine learning approaches used to learn the security needs from the various data sources across the industry. Focus areas of the paper are discussed in the next section, followed by the understanding of threat modeling. The application of risk assessment with threat modeling is explored in the next section. The data collection-related process is discussed in the subsequent section. Experiments are built on understanding how threat prediction can be introduced into the conventional threat modeling approach. The outcome of these experiments is validated with the available best approaches and their results. The Paper is wrapped up with a discussion on weaknesses in the work and possible future prospective areas.

II. LITERATURE REVIEW

In work [5], neural networks, deep learning techniques, and ensembles were explored in cyber security. Cyber security areas of intrusion detection, prediction of cyber-attacks, and malware identification are targeted areas. This paper provides a reference point for cyber security professionals in deep learning. This work highlights the need for further exploration to make the algorithms more efficient based on the specific data under study [6], [7]. Challenges in data collection are also highlighted as the area that needs focus. This paper explores a variety of the deep learning approach in the space of cyber security. However, these approaches are not covered by software development processes. We leverage some of the learning from this paper and explore them for software development processes in our work. In work, [8], machine learning and deep learning approaches are explored to tackle the security issues bothering big data. Due to considerable growth in data volumes, there are vulnerabilities for the security threats to hamper the system. In [9] and [10], an exploration of taxonomy around the threat modeling approach was achieved with the machine learning and deep learning approaches. However, the practical implementation in software development eco-systems was not achieved. In [11], the growing application of machine learning and the possible vulnerabilities introduced into the system were explored. The study covers the threat model for machine learning and explores the attack involved and the defenses that would be needed. This paper takes away some of the learning around threat modeling. Work attempts to bring perspective around model accuracy, complexities, and resilience that needs attention based on its operating environment [12], [13].

Work [14] explores machine learning capabilities for cyber security. Machine learning capabilities to identify advanced threats and targets in infrastructure vulnerabilities, organization profiling, and other exploits are explored. With the inability of the traditional malware handling approaches, these new capabilities come in handy. From this work, we take away the key insights of applying machine learning in the cyber

security space and re-use and test it in the software development space. Work [15] focuses on providing insights into threat modeling. Exploration done on Microsoft's threat modeling is used as a base to offer insights into an effective threat modeling approach. Work [16], [17] explores threat modeling applications in agile software development processes with Microsoft's STRIDE approach. Practical challenges facing the industry are explored and validated with the challenges highlighted in the literature. Some of the key challenges are seen during the identification of assets stage and how the post-threat modeling exercise is implemented.

In [17], a variety of vulnerabilities were studied. IT helped to understand the granular details of the vulnerability. This knowledge helps to build the datasets for our experiments. This knowledge also helps to enhance the construction of machine learning experiments. In work, [18], automation of threat modeling is focused. The focus is to reduce the effort involved in the threat modeling by leveraging the available data [19], [20]. This work helps to understand the thought process behind the threat modeling framework. This learning helped build the threat modeling framework that can work with other security-related frameworks. Lack of context is another challenge faced by the prediction models. This lack of context is due to the lack of domain knowledge being considered in the modeling process. Authors try to bring in an ontology framework to improve the conceptual modeling. In work [21], authors introduce threat identification as part of the software development lifecycle. The idea here is to reduce the need for educating software development experts on security knowledge. The proposed approach looks at analyzing the design of the software to explore risks and threats to the system. Authors introduce an identification tree named a new data structure approach for detection of threats to [22]. The mitigation tree approach is utilized for the description of countermeasures. These methods provide a guided approach for risk assessment across the software development lifecycle.

III. RESEARCH GAPS

Based on the literature review done, we see that there are the following research gaps. Focus on learning the structure of customer requirements in agile development methodology from a security perspective which is missing now. Utilizing the construct of security categories from the industry data to derive the implicit security needs of the customer. These elements are essential to bridge the conventional threat modeling, and risk assessment approaches with machine learning capabilities. Integrating the customer, industry, and software processes data sources to learn the security needs is not addressed appropriately and needs deeper exploration. This approach helps to have a comprehensive view of the proposed framework's security vulnerabilities.

Deep learning advancements in cyber security are another area that needs attention. Improvisation of algorithms based on the data eco-system can provide good leverage for the research in software development practices improvisation. Deep learning approaches to the exploration of software development space need more effort. Deeper work is needed on addressing security issues in software development. Software system threat modeling automation needs more focus. Deeper work is needed on associating context and domain knowledge for modeling the information. A stronger association of risk assessment and threat modeling good practices would be important to leverage the power of both the framework.

IV. KEY THEMES OF THE STUDy

In work [3], we have outlined an efficient system that can facilitate information management in software development processes. The intent is various sources of the software development process data and ways to model them to supply it as helpful information [23], [24]. The overall objective of this integrated information system is to combine the information in the areas of customer conversation, industry best practices, and internal software development processes. Figure 1 depicts the system outline of the conceptualized Integrated Information Management system to tackle security vulnerabilities.

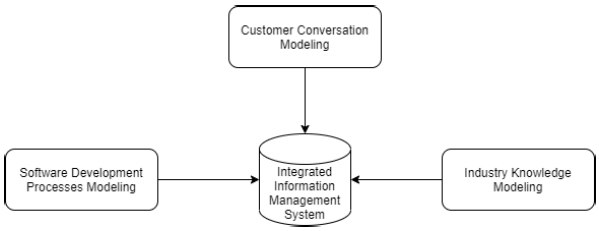

FIGURE 1. Conceptualized Integrated Information Management system to handle security vulnerabilities.

Under the module of industry knowledge modeling, threat modeling concepts are studied. This study will reduce the inefficiencies prevalent in the threat modeling exercise. Machine learning approaches are explored to build predictability into the exercise. The information available in the industry around the threats and vulnerabilities can be leveraged to provide the information needed when it matters. A combination of vulnerability identification from customer conversation, internal software development processes, and industry knowledge will help build a robust information system.

V. THREAT MODELING APPROACH

Cyber security risks that challenge the software system make it essential for the industry to take a proactive step towards tackling it effectively. The complexity of threat modeling makes it less effective when it is implemented. The best approach is to start with simple steps and build it [25], [26]. Constructing the software systems boils down to the requirements that specify the features needed, acceptance criteria from the customer, and a technical breakdown of the requirements. A specific standardized approach to threat modeling is missing which makes the situation harder [27].

Technical risks would be a good starting point as they are particular to the software system, like the ones around missing security control in the software. Since the software system's structure is well within the control, it will be easier to handle. Making risk identification a collaborative effort goes a long way in maintaining effectiveness in the system [26]. Agile methodology has a nice setup of the team structure where the product owner, system analyst, developer, tester, architect, and scrum master form a scrum team that intends to deliver the value-based product to the customer. This mix of expertise across the value chain can be a good setup for collaborative threat assessment. Cyber security risks go beyond ticking the checklist and making sure the business risks are kept under check [28], [29].

Breaking the system into smaller components to start the analysis will be a good starting place. This specific focus helps to take action more frequently and see the progress. This iterative approach of threat modeling will help get everyone's involvement rather than a complex analysis done at the beginning of the project [30], [31]. Exploration, brainstorming of the threats, and prioritizing and fixing the threats are the simple starting points to implement. Deciding on the stakeholders needed for the threat exploration is essential. Frequency to be agreed upon for the threat exploration session. It is always better to have a face-to-face session with the people involved rather than an online session. This aspect has been our experience while conducting brainstorming sessions with the software development team for threat modeling analysis. Figure 2 shows the simple format to identify the threat in the system.

As discussed earlier, prioritizing, and taking up the important work to time box the exercise is essential. This focus helps to maintain the healthy progress on threat modeling exercise. The latest features that are worked upon, any identified security feature, services that are collaborating with other services, and technical security debt are good areas to start focusing upon [32].

VI. COMBINING RISK ASSESSMENT APPROACH AND THREAT MODELING

The risk assessment approach that we have been practicing in our company is based on information security assets. We look at confidentiality, integrity, and data availability as a primary focus areas for our security risk assessment. Based on the probability of occurrence of events that compromise these three factors and their impact, we arrive at the risk level. Based on these risk levels, risk mitigation actions are devised. Risk mitigation will be around security controls needed to manage those risks. Threat modeling for security risks focuses on the technical risks involved in software systems. It focuses on all phases of the software development lifecycle, including requirements gathering, design, construction, and testing. Threats and vulnerabilities hampering the software systems are used as a base in this assessment. We integrate the risk assessment and threat modeling approach with data analytics approaches in our work. In the further part of this section and subsequent sections, we propose our approach. In the first stage of this risk assessment approach-based threat modeling, all the components of the software system and processes are to be listed. For example, network and communication-related components, software components, and other similar areas. In the next phase, impact analysis is conducted. To start with this exercise, initial impact analysis can be expert judgment-based. Later a database can be set up to track the events that will feed into automated impact analysis [33]. Impact analysis includes three components, what is the result of compromise on the confidentiality of the data, integrity of the data, and availability of the data. We have used this approach of risk assessment based on confidentiality, integrity, and availability in our company and have observed that it provides a comprehensive view of the security risks and their impact. Based on these three components, impact value can be derived. The organizational database can be created to collate the experience and events, which will help understand the impact of compromise of data from all three perspectives referred to [34]. This database can be an ongoing repository that helps build the knowledge base for impact analysis. Impact value of confidentiality, availability, and integrity can be provided with a range of values based on their impact on the customer. Based on the combination of values of these three parameters, the final impact value can be arrived at in this work [35].

Data collected in the organizational database can predict the impact value. All the attributes associated with the identified components can be put together to model the impact based on confidentiality, integrity, and availability. Alternatively, any other parameters would help build a threat modeling system. Impact value can also be directly derived from the attributes associated with the target components. In the next stage, based on the categories of the components, possible threats and vulnerabilities that would impact the components can be listed. This listing can be based on the organization's historical data or industry knowledge. To build a historical experience-based list, it is essential to have a process that helps capture all the threats that have hampered software system components and vulnerability in the system that has led the threat to exploit. If we take the technical failure of the software components as the threat, it would be caused by vulnerabilities like inadequate business continuity management, inadequate system monitoring, insufficient user testing, and others. Using industry data, we can model the threat and vulnerabilities based on the information related to software system failures and the causes.

The next part of the information needed is the probability of occurrence of these vulnerabilities. This information will help to assess the risk level of the failures further. The probability of occurrence can be provided on a high, medium, or low scale based on the number of times those events have occurred in the past. This tracking needs a system to capture all the events associated with the software system deficiencies from related to confidentiality, integrity, and availability. As discussed earlier, these three factors or any

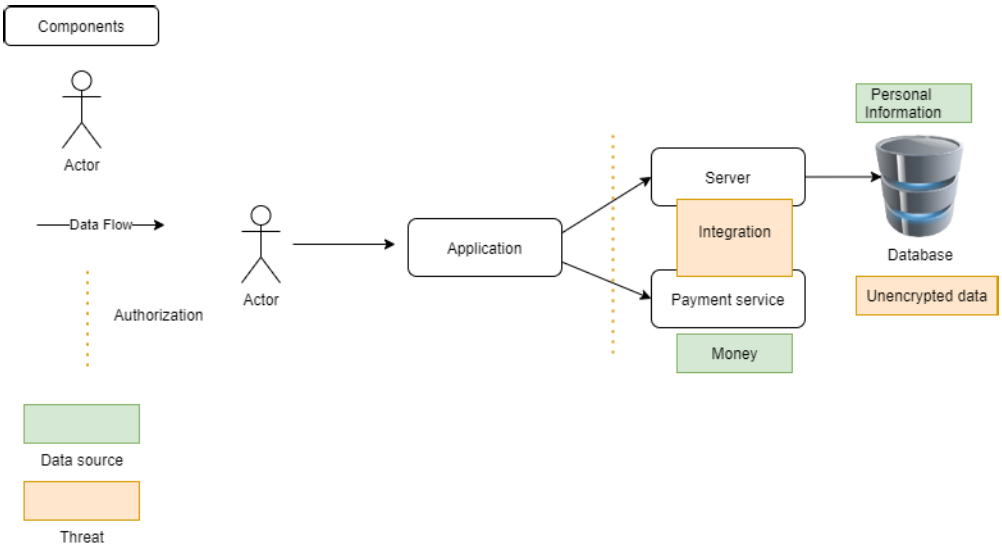

FIGURE 2. Depiction of simple threat modeling.

other factors that are relevant to the system can be considered. While modeling the threats is done in this approach, impact evaluation and occurrence evaluation can be combined to obtain the final risk levels of the components. The final risk level is a combination of impact value and probability of occurrence of the vulnerabilities in the past. Components can be subjected to the study of controls needed based on the risk level. Controls needed can be based on the threat type and its vulnerabilities type. Control information can be derived from industry knowledge databases. Control refinement can be carried out based on the residual risks after implementing the control. So, the system can be made in real-time where it captures all the information periodically and recalibrates the system for its risk value and controls applied. Based on future events, this system calibrates itself and provides direction for further strengthening. Figure 3 depicts the outline of a risk assessment-based approach for threat modeling. We consider Impact value $= I_v$, Impact factors identified $= T_f$, Risk value-R_v, Probability of occurrence $= P_O$.

$$I_v = Avg(T_f). \tag{1}$$

$$R_v = I_v * P_O. \tag{2}$$

VII. DATA COLLECTION

Threat modeling for the software development processes can be done concerning the information captured in the software development processes. Threat categories will have multiple CWE (Common Weaknesses Enumeration) under them. CWE is a community-developed list of software and hardware weakness types [36]. In the software development processes with the Agile framework, requirements are documented in the form of user stories, which are further broken down into tasks. TFS (Team Foundation Server) is used as ALM (Application Life-cycle Management) tool. Test cases are created to cover all the expected testing scenarios. Any issues identified during the software development are tracked as defects and addressed.

Tasks are linked to the user stories; test cases are also linked to the user stories. Any defects found during testing are linked to test cases. These linkages help to maintain traceability. Leveraging these work items' traceability, requirements can be mapped to defects that are related to security; additional security-related issues can be traced to the CWE. Expert involvement is needed to map the security issues to CWE. Required training and knowledge sharing must be enabled for this process. Building a model around the patterns of software requirements, to security issues to associated CWE will help understand the possible threats that would hamper the software system. In this data collection approach, linkages between these work items are leveraged. CWE mapping done with the involvement of software development experts is leveraged for the modeling. The idea is to build a prediction engine that can predict possible CWEs that would get resulted when a customer requirement is being worked on. This identification provides an opportunity for the software development teams to engage the security controls much earlier in the process.

All the work items from TFS are extracted, including user stories, tasks, test cases, and defects. All the work items that have reference to CWE are selected. Parents' work items for these work items are also collected to trace back to original requirements. CWE ids are separated from the text content, which will act as a label for the text descriptions. All the information available across work items in the form of their title and description with CWE being referred are extracted to create a data source that has text description and the CWE

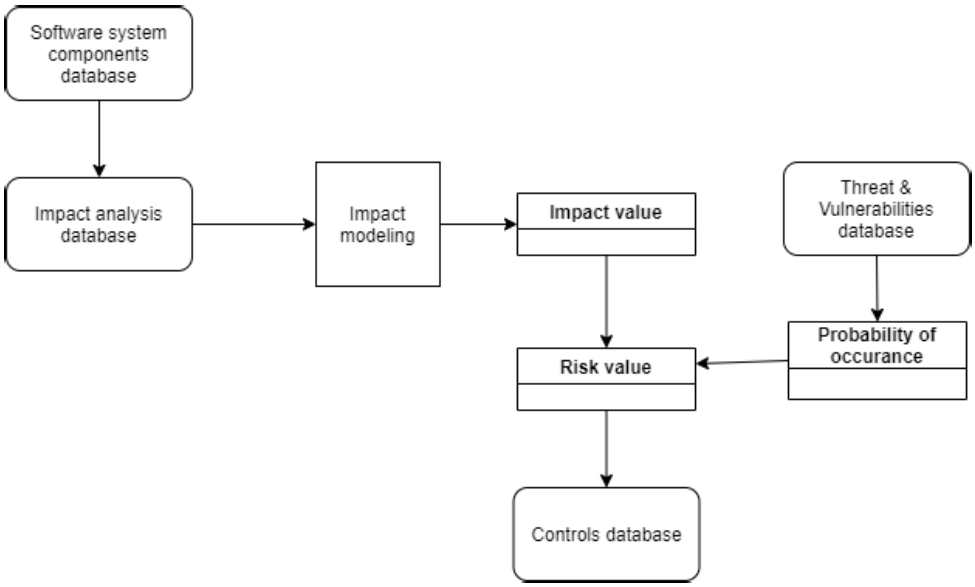

FIGURE 3. Risk assessment-based threat modeling.

mapping. From the data extracted, 1458 text data descriptions are available that are mapped to 64 different categories of CWEs.

VIII. PREDICTION MODEL FOR THREAT ASSESSMENT

The prediction model intends to categorize the customer requirements into respective CWE categories. Once the model is built around this content, it will be possible to map the new requirements coming from customers to their possible CWE and predict the potential threats that may hamper the software system. Based on the CWEs mapped, security controls can be devised to tackle threats to software systems.

A. MULTI-CLASS CLASSIFICATION APPROACHES

In the first phase of this exercise, a text description is subjected to TF-IDF (Term Frequency Inverse Document Frequency) for the vectorization process. Logistic regression, Random forest classifier, Multinomial NB (Naïve Bayes), and Linear SVC (Support Vector Classifier) are used for classification modeling. Random forest classifier is tuned with n_estimator of 200 and max_depth of 3. Cross-validation of 5 is chosen for the modeling. Table 1 shows the results of the first round of modeling. Table 1 shows the results of the first round of modeling.

TABLE 1. Results of first round of modeling.

Model name	Accuracy
Linear SVC	47.20%
Logistic Regression	39.85%
Multinomial NB	30.25%
Random Forest Classifier	23.52%

Linear SVC showed a precision of 51%, recall of 48%, and F1 score of 46% on a weighted average scale. Performance is not up to mark.

TABLE 2. Model results with parameter details.

Models (Classifier)	Parameters	F1 score
Random Forest XGB Logistic Regression	Default parameters	11.98%
Logistic Regression	Default parameters	14.38%
XGB	Default parameters	38.35%
RandomForest	Default parameters	37.67%
Logistic Regression	penalty='l2', class_weight=None, random_state=100, solver='lbfgs', warm_start=False, l1_ratio=0	13.01%
Logistic Regression	penalty='elasticnet', class_weight='balanced', random_state=100, solver='saga', warm_start=True, l1_ratio=1	4.79%
XGB	max_depth=3, n_estimators=100, verbosity=1, objective='binary:logistic', booster='gbtree', gamma=0, reg_alpha=0, reg_lambda=1, random_state=100	36.64%

In this section, we try to build ensemble models. Preprocessing of the data is conducted with Beautiful Soup and tqdm libraries. Also, TensorFlow Kera's preprocessor is used to tokenize the natural language data used as input. The text-to-sequence method is used for this purpose. Input data is split into train and test components with 80% data for training and 20% data for testing. This round tries the Random Forest Classifier, XGB (XG Boost) classifier, and Logistic Regression classifier. All three models' outputs are averaged in this method to obtain better performance from the ensemble model. Table 2 shows the performance of various models in terms of F1 score and the parameters that are used. This experiment shows that XGB is the best among all the parameters but not good enough. Further ensemble

methods are explored based on the feasibility study of the software development work items and the details of the domain experimented on. Multinomial NB (Naïve Bayes), Decision Tree Classifier, K Neighbors Classifier, Linear SVC (Support Vector Classifier), and Random Forest Classifier are used. These algorithms are started with averaging methods prediction. The F1 score metric is used for the evaluation of performance. Table 3 shows the performance of various models in terms of F1 score and the parameters used.

Table 4 lists out the parameters chosen when all the models were run together. Since Random Forest Classifier and KNeighbors Classifier were relatively better, they were run together under averaging method, but their actual performance was reduced by 3%. Under the max voting method also the performance is only about 31%.

In this section, ensemble, deep learning models that are appropriate to model the data from software development processes focusing on the security of the software are shortlisted. An attempt has been to classify the content captured in software development work items into security-related content mapped to respective CWE. This mapping will help to call out the possible threats hidden in the system. Spacy library from NLP (Natural Language Processing) is used for data processing. Training and testing data of 70% and 30% are constructed for the experiment. Kera's preprocessing library text tokenizer and pad sequencer are applied. A pretrained model glove with 200 dimensions is utilized for the generation embedding matrix used for training the model.

B. CNN (CONVOLUTIONAL NEURAL NETWORK) STATIC

CNN static algorithm architecture includes layers of CONV1D, BatchNormalization, Activation, and GlobalMaxPool1D being concatenated. Dropout is kept at 50%, followed by a dense layer of 512 units and 'relu' activation. The output layer is a dense layer with a 'softmax' activation function. CNN static model is compiled with loss function of 'categorical_crossentropy,' optimizer 'adam,' and batch_size of 128 with a function written to compute top three accuracies. CNN static model is created with the 'Model' function from Keras.model library. This model is further run with ''fit_generator'' to feed data in sequential mode. The top 3 accuracies show a performance of 50.68%. Performance on training and hold-out data set over the epochs in terms of the loss is depicted in figure 4. The hold-out set cannot close on the training dataset in terms of the loss value. 80% and 20% split of training data is a general guideline. In ensemble and deep learning models, we want to experiment with different training and testing data spilled. However, this did not make much of a difference at the end of the experiment.

C. CNN DYNAMIC

To make the CNN network dynamic, in the embedding_layer creation, the parameter 'trainable' is kept to 'True' so that the training happens dynamically. The architecture of the network remains the same as the CNN-Static network. Model building and compilation stay the same. The top 3 accuracies

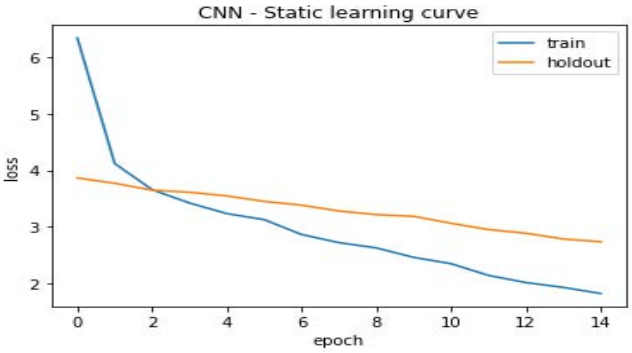

FIGURE 4. Training and hold-out set performance trend over epoch in terms of the loss in case of CNN-static.

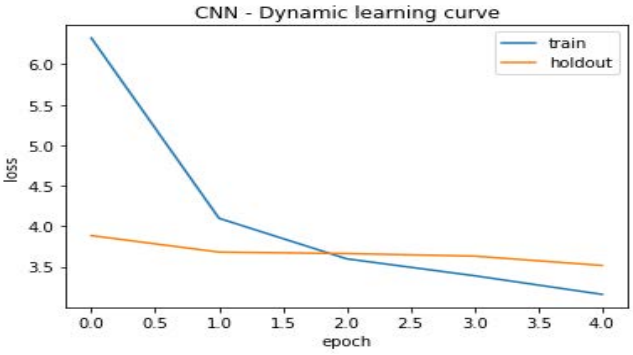

FIGURE 5. Training and hold-out set performance trend over epoch in terms of the loss in case of CNN-dynamic.

show a performance of 31.76%. Performance on training and holdout data set over the epochs in terms of the loss is depicted in figure 5. Though the holdout data set is close to training dataset performance on loss. Overall, they cannot do better compared to CNN- Static.

D. DATA PROCESSING, TRANSFORMATION, AND MODELING

In this section, data pre-processed with NLP's spacy library is used. Tfidf (Term frequency-inverse document frequency) vectorizer is used for the data vectorization process. Now the data is subjected to models Logistic Regression, Random Forest Classifier, and Linear SVC. A cross-validation value of 5 is chosen for the processing. Random Forest uses ''n_estimators'' of 300 and ''max_depth'' of 3. Accuracies of the models are shown in figure 6.

Table 5 depicts model performance in terms of accuracy. Even modifications in the data processing models do not significantly improve their performance. Based on the literature review, some models that have shown good performances for the classification problems will be explored here. Exploration will look for compatibility of these models for the data used here, coming from software development processes that follow Agile methodology and serve the insurance domain business.

TABLE 3. Model performance and parameter details.

Model	Parameter	F1-score
Multinomial NB	Default parameters	3.08%
Decision Tree Classifier	Default parameters	18.83%
K Neighbors Classifier	Default parameters	32.53%
Linear SVC	Default parameters	10.61%
Random Forest Classifier	Default parameters	32.19%
All models	Tunned parameters	41.09%
Random Forest Classifier K Neighbors Classifier	Default parameters	29.45%

TABLE 4. Parameters chosen for various models used during ensemble run of all models.

Model	Tuned Parameter
Decision Tree Classifier	criterion = "entropy", random_state = 100, max_depth=None, min_samples_leaf=5, splitter='random'
KNeighbors Classifier	n_neighbors=5, weights='distance', algorithm='auto', leaf_size=30, p=1, metric='minkowski'
Linear SVC	penalty='l1', loss='squared_hinge', dual=False, tol=0.0001, C=1.0, multi_class='ovr', fit_intercept=True, intercept$_s caling$ = 1, $class\_weight$ = $'$ $balanced'$, $random\_state$ = 100, max_iter = 1000
Random Forest Classifier	n_estimators=200, criterion='entropy', min_samples_split=2, min_samples_leaf=1, random_state=100

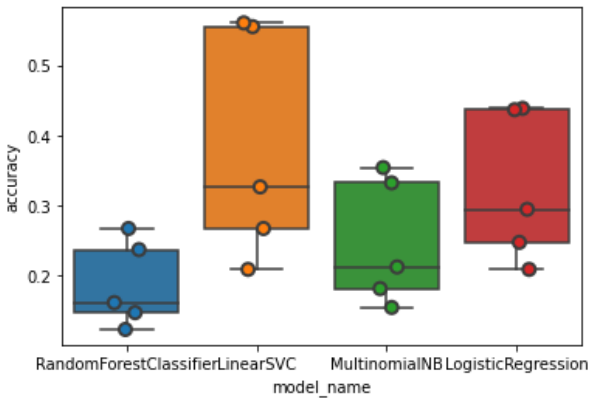

FIGURE 6. Accuracies of the models run in this section.

Naïve Bayes, KNearest Neighbor, Support Vector Machine, Random Forest, Decision Tree, and ensemble classifiers are the ones to be explored. There was no significant improvement in the performance so raw data will be used directly without the NLP space-based processing done in the previous section. Tfidf vectorizer will be used on this data for vectorization purposes. The cross-validation value is maintained at 5. Table 6 shows the model parameters and their performance.

Except for the improvement of the Linear SVC model, where accuracy improved to 47.61%, the rest of the models are still not doing well. Ensembling the best models among these will be explored for the data used here. Stacking ensemble modeling is tried in the next section. There will be level 0 and 1 models; a stacking classifier combines the models

TABLE 5. Model performance in terms of accuracy.

Model	Accuracy
Linear SVC	38.34%
Logistic Regression	32.51%
Multinomial NB	24.69%
Random Forest Classifier	18.72%

from levels 0 and 1. The logistic regression model is used as a level 1 model, and the rest are configured as a level 0 model. Data processing is kept to raw data being processed with the Tfidf vectorizer.

Repeated Stratified KFold method from sklearn's model selection library is used to configure the 'cv' parameter for the modeling. Parameter set for 'RepeatedStratifiedKFold' are 'n_splits of 10 and 'n_repeats' of 3. 'cross_val_score' method from sklearn's model selection library is used to generate scores to evaluate the model. This method uses parameters, 'model,' 'input data,' 'label data,' 'scoring methods,' and 'cv' value. The scoring method used is accuracy, and the cv value gets generated from the 'RepeatedStratifiedKFold' method. Performance of various models with their parameter configuration is provided in table 7.

In the first round of stacking with all models, accuracy was poor at 0.2%. As per the literature review, decision tree, neighbors, and Logistics regression has performed well in a similar setup. The stacking of these models improved the performance to 44.6%. Individual model performances of Logistic Regression, Decision Tree classifier, and Support Vector Classifier show better performance. However, removing KNeighbors and adding a Support Vector

TABLE 6. Model parameters and its performance in accuracy.

Model	Parameter	Accuracy
Random Forest Classifier	n_estimators=200, max_depth=3	24.01%
Multinomial Naïve Bayes	Default parameters	30.25%
KNeighbors Classifier	n_neighbors=5, metric='euclidean'	37.24%
Linear SVC	Default parameters	47.61%
Decision Tree Classifier	criterion = "gini", max_depth=3, $\text{min\_samples}_leaf = 5\%$	13.92%

TABLE 7. Model's parameter configuration and performance.

Models	Parameters	Accuracy	Standard Deviation
Logistic Regression	Default parameters	41.7%	2.9%
KNeighborsClassifier	Default parameters	38.6%	3.5%
Decision TreeClassifier	Default parameters	46.7%	3.7%
Support Vector Classifier	Default parameters	43.5%	3.4%
GaussianNaïve Bayes	Default parameters	37.1%	4.1%
Stacking (All Models)	Default parameters	0.2%	0.4%
Stacking (Decision Tree, KNeighbors, Logistic Regression)	Default parameters	44.6%	3.2%
Stacking (Logistic Regression, Decision Tree classifier and Support Vector Classifier)	Default parameters	0.2%	0.4%
Stacking(Logistic Regression, Decision Tree classifier, Support Vector Classifier and Multinomial Naïve Bayes	Default parameters	44.9%	2.8%
Multinomial Naïve Bayes	Default parameters	28.6%	2.6%
XGBoost Classifier	Default parameters	49.4%	3.9%
LogisticRegression	class_weight= 'None', max_iter= 200, multi_class= 'ovr', penalty= 'none', solver= 'saga', random_state=100	50.6%	3.9%
Decision Tree Classifier	criterion='gini', max_features= 'auto', splitter= 'best'	41.3%	3.9%
Stacking(LogisticRegression and Decision Tree Classifier)	Tuned as per parameters in previous 2 rows	38.2%	3.3%

classifier reduced stacking performance back to 0.2%. This performance indicates the earlier combination was best. With the best performers, Multinomial Naïve Bayes is added, and performance slightly improved to 44.9%, whereas Multinomial Naïve Bayes performed at 28.6%. XG Boost classifier took the performance to 49.4%, but this is computationally expensive, so it would not be feasible. Logistic regression and Decision tree classifier were fine-tuned with grid search CV, and performance was 50.6% and 41.3%, respectively, but stacking performance was reduced to 38.2%.

The original database had data across 63 CWEs categories. Many of the CWEs categories had only a few data points under them. This state resulted in an imbalanced dataset, and models were poorly performing. The top 20 CWEs were more prevalent upon discussion with application development experts. These top 20 CWEs frequency of occurrence was also observed to be high. Only 20 of the most occurring CWE were shortlisted based on expert input. To improve the prediction performance, more data was collected from across other programs in the company. Three thousand two hundred fourteen data points from across multiple programs were collected for the 20 CWEs that were shortlisted. Among all the above experiments conducted, the stacking model of the Decision Tree classifier, KNeighbors classifier, and Logistic Regression showed the best performance with 77.7% accuracy and a standard deviation of 2.5%. The decision tree classifier and KNeighbors were used at level 0, and the Logistic Regression model was used at level 1 in the stacking.

Based on the variation of the occurrences of the CWEs in the future, modeling must be fine-tuned to cover more CWEs. Collaboration with experts is to be continued to study the outcome of the current model. The experts must validate predicted labels. Labeling of the data into appropriate CWEs must be improved during validation. This ongoing effort will help improve the prediction engine to a much better level.

Evaluation against the state-of-art: Work [37], utilizes SMOTE, SVM with RBF kernel, and logistic regression approaches utilizing Recordings of meetings between developers and customers from a software development company in the United States. Here they explore a classification approach to figure out security vulnerabilities. They have recorded the results of Precision at 70.8% and Recall at 18.3%.

Work [38] explored LDA and SVM approaches with Stack Overflow dataset for classification of the security vulnerabilities from the data. The following results were produced. For LDA, Precision was 70.33%, Recall was 77% [39], [40]. For SVM (Support Vector Machine), Precision was at 72%,

and Recall was at 77% [41]. Among all the above experiments conducted, the stacking model of the Decision Tree classifier, K-Neighbors classifier, and Logistic Regression showed the best performance with 77.7% accuracy and a standard deviation of 2.5%. Decision Tree classifier and K-Neighbors were used at level 0, and the Logistic Regression model was used at level 1 in the stacking. In comparison to these best works, we would achieve better performance with a precision of 76% and recall of 79%.

Methods used in the paper are briefed by starting with the background of existing methods. We start from the ''Understanding threat modeling'' section, where there is an exploration of automating some of the sub-processes using machine learning. In the next section, ''Risk assessment approach for threat modeling,'' there is an exploration of combining the conventional risk assessment method with the threat modeling approach. This approach helps to leverage the best of both approaches. In the section ''Threat prediction modeling,'' the core proposal of our work is detailed.

In this section, 'Data Transformation and Modeling,' we did a detailed study of the machine learning algorithms that will fit in our architecture. Starting from basic machine learning algorithms to more advanced algorithms were explored. A comparison of the performance of various experiments was conducted. Models' parameter tuning and the best combination of the parameters are explored in detail to arrive at the best combination of parameters for models working well. Table 7 provides a comprehensive view of all the models, parameters, and performance. We also provide the best outcome of the performance on our dataset.

Some of the methods used are as follows. TF-IDF is the famous approach for weighing the terms in Natural Language Processing. Logistic Regression, Random Forest Classifier, and Support Vector Classifier are basic machine learning approaches used for classification problems. Beautiful Soup is a python library utilized for web scrapping from XML and HTML pages. Tensor Flow is an open-source artificial intelligence library that uses data flow graphs to build models. Keras is a neural network library that provides high-level APIs for building and training models. XG Boost stands for eXtreme Gradient Boosting and is a supervised learning library with parallel processing capabilities. CNN is an artificial neural network for processing image data. Ensemble classifiers help to improve machine learning outcomes by combining various models.

IX. CONCLUSION

In this exploration, the focus was to build a quantitative threat modeling approach. It is essential to use the knowledge prevalent in the software development processes and from across industries. The information available from the industry has been overwhelming for the software development team to leverage. Approaches discussed in this paper will help make this information available as and when needed. Security challenges are faced depending on the business domain in which the industry is operating. In this study title, the insurance business domain is the focus area. The title insurance business domain is unique from other branches of the insurance business. Software development processes following the agile development model also provide different set-ups in which security improvements can be focused. Ongoing calibration of this system is needed to strengthen the system.

It is essential to calibrate the data store for identifying the right CWEs in the software development processes. Identification of all security-related events is also another crucial aspect. All these calls for appropriate education of the software development communities on security practices.

These proposed systems need to adapt and learn from dynamic changes in the industry. There are new vulnerabilities that are discovered in the industry regularly. The system and people need to be up to date on these dynamics of security issues. This work contributes to establishing an integrated and automated approach for software threat modeling. Studies of conventional threat modeling and security risk assessment are conducted, and the best of both are brought together with machine learning approaches. Machine learning approaches are customized to get better results than other related work done earlier. We have demonstrated a machine learning architecture appropriate to the subject under study and one that shows promising results with available data.

X. FUTURE STUDY

This study is limited to looking at software systems without looking at the category to which the software system belongs. Visibility into the class or category of the software system and study-specific to those classes can make the outcome more effective. This area is a good one for future research. Modeling conducted in the study is generic; exploring the machine learning models that can leverage the contextual information from the data can make the experiments further stronger. This area must be developed in future studies. The agile software development model and security-related controls in software development may have varied objectives. These objectives are not analyzed concerning each other as part of our study. Putting these together and aligning the work will help to optimize the framework further and needs focus in future studies. Our work is also limited to building larger datasets to leverage the capabilities of the deep learning methods. This area can be a focus for future studies. Imbalanced datasets are another area that needs focus regarding security-related data and machine learning approaches. There are many categories of security threats that are less frequent, but when they occur, they will impact badly; this needs to be explored further. Unsupervised anomaly detection methods would help bring in more efficiency in this research area and need exploration.

XI. DECLARATIONS

A. AVAILABILITY OF DATA AND MATERIALS

The data used to support the findings of this study are available from the corresponding author upon request.

B. COMPETING INTERESTS

This article does not contain any studies with human participants performed by any of the authors. There is no conflict of interest between authors.

REFERENCES

[1] N. G. Eapen, A. R. Rao, D. Samanta, N. R. Robert, R. Krishnamoorthy, and G. H. Lokesh, "Security aspects for mutation testing in mobile applications," in *Cyber Intelligence and Information Retrieval* (Lecture Notes in Networks and Systems), J. Manuel, R. S. Tavares, P. Dutta, S. Dutta, and D. Samanta, Eds. Singapore: Springer, 2022, pp. 17–27.

[2] P. K. Singh, "Data with non-Euclidean geometry and its characterization," *J. Artif. Intell. Technol.*, vol. 2, no. 1, pp. 3–8, Dec. 2021.

[3] R. R. Althar and D. Samanta, "The realist approach for evaluation of computational intelligence in software engineering," *Innov. Syst. Softw. Eng.*, vol. 17, pp. 17–27, Jan. 2021.

[4] A. Balakrishna and P. Mishra, "Modelling and analysis of static and modal responses of leaf spring used in automobiles," *Int. J. Hydromechatron.*, vol. 4, no. 4, pp. 350–367, 2021.

[5] I. H. Sarker, "Deep cybersecurity: A comprehensive overview from neural network and deep learning perspective," *Social Netw. Comput. Sci.*, vol. 2, no. 3, pp. 1–16, May 2021.

[6] R. R. Althar, D. Samanta, D. Konar, and S. Bhattacharyya, *Software Source Code*. Berlin, Germany: De Gruyter, July 2021.

[7] S. C. Mondal, P. L. C. Marquez, and M. O. Tokhi, "Analysis of mechanical adhesion climbing robot design for wind tower inspection," *J. Artif. Intell. Technol.*, vol. 1, no. 4, pp. 219–227, Sep. 2021.

[8] R. Gupta, S. Tanwar, S. Tyagi, and N. Kumar, "Machine learning models for secure data analytics: A taxonomy and threat model," *Comput. Commun.*, vol. 153, pp. 406–440, Mar. 2020.

[9] R. R. Althar and D. Samanta, "Building intelligent integrated development environment for IoT in the context of statistical modeling for software source code," in *Multimedia Technologies in the Internet of Things Environment* (Studies in Big Data), R. Kumar, R. Sharma, and P. K. Pattnaik, Eds. Singapore: Springer, 2021, pp. 95–115.

[10] L. Li, Q. Dong, D. Liu, and L. Zhu, "The application of fuzzing in web software security vulnerabilities test," in *Proc. Int. Conf. Inf. Technol. Appl.*, Nov. 2013, pp. 130–133.

[11] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, "Towards the science of security and privacy in machine learning," 2016, *arXiv:1611.03814*.

[12] A. R. Rao and D. Samanta, "A real-time approach with deep learning for pandemic management," in *Healthcare Informatics for Fighting COVID-19 and Future Epidemics* (EAI/Springer Innovations in Communication and Computing), L. Garg, C. Chakraborty, S. Mahmoudi, and V. S. Sohmen, Eds. Cham, Switzerland: Springer, 2022, pp. 113–139.

[13] T. V. Hahn and C. K. Mechefske, "Self-supervised learning for tool wear monitoring with a disentangled-variational-autoencoder," *Int. J. Hydromechatron*, vol. 4, pp. 69–98, Mar. 2021.

[14] J. B. Fraley and J. Cannady, "The promise of machine learning in cybersecurity," in *Proc. SoutheastCon*, Mar. 2017, pp. 1–6.

[15] I. Williams and X. Yuan, "Evaluating the effectiveness of Microsoft threat modeling tool," in *Proc. Inf. Secur. Curriculum Develop. Conf.*, Oct. 2015, pp. 1–6.

[16] D. S. Cruzes, M. G. Jaatun, K. Bernsmed, and I. A. Tøndel, "Challenges and experiences with applying Microsoft threat modeling in agile development projects," in *Proc. 25th Australas. Softw. Eng. Conf. (ASWEC)*, Nov. 2018, pp. 111–120.

[17] K. Goseva-Popstojanova and J. Tyo, "Experience report: Security vulnerability profiles of mission critical software: Empirical analysis of security related bug reports," in *Proc. IEEE 28th Int. Symp. Softw. Rel. Eng. (ISSRE)*, Oct. 2017, pp. 152–163.

[18] M. Välja, F. Heiding, U. Franke, and R. Lagerström, "Automating threat modeling using an ontology framework," *Cybersecurity*, vol. 3, no. 1, pp. 1–20, Dec. 2020.

[19] R. R. Althar and D. Samanta, "Application of machine intelligence-based knowledge graphs for software engineering," in *Methodologies and Applications of Computational Statistics for Machine Intelligence*. Hershey, PA, USA: IGI Global, 2021.

[20] Y. Xu, Y. Li, and C. Li, "Electric window regulator based on intelligent control," *J. Artif. Intell. Technol.*, vol. 1, no. 4, pp. 198–206, Sep. 2021.

[21] E. E. Heymann César and B. P. G. Miller Ruiz, "Automating threat modeling through the software development life-cycle," *XXIII Jornadas de Paralelismo*, pp. 21–38, May 2012.

[22] L. Williams, G. McGraw, and S. Migues, "Engineering security vulnerability prevention, detection, and response," *IEEE Softw.*, vol. 35, no. 5, pp. 76–80, Sep./Oct. 2018.

[23] P. K. Kudjo, J. Chen, S. A. Brown, and S. Mensah, "The effect of weighted moving windows on security vulnerability prediction," in *Proc. 34th IEEE/ACM Int. Conf. Automated Softw. Eng. Workshop (ASEW)*, Nov. 2019, pp. 65–68.

[24] D. Jie, G. Zheng, Y. Zhang, X. Ding, and L. Wang, "Spectral kurtosis based on evolutionary digital filter in the application of rolling element bearing fault diagnosis," *Int. J. Hydromechatron.*, vol. 4, no. 1, pp. 27–42, 2021.

[25] O. Alhazmi, Y. Malaiya, and I. Ray, "Security vulnerabilities in software systems: A quantitative perspective," in *Data and Applications Security XIX* (Lecture Notes in Computer Science), S. Jajodia and D. Wijesekera, Eds. Berlin, Germany: Springer, 2005, pp. 281–294.

[26] L. Allodi, M. Cremonini, F. Massacci, and W. Shim, "Measuring the accuracy of software vulnerability assessments: Experiments with students and professionals," *Empirical Softw. Eng.*, vol. 25, no. 2, pp. 1063–1094, Mar. 2020.

[27] G. Jabeen and L. Ping, "A unified measurable software trustworthy model based on vulnerability loss speed index," in *Proc. 18th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./13th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Aug. 2019, pp. 18–25.

[28] W. Xiong, M. Gülsever, K. M. Kaya, and R. Lagerström, "A study of security vulnerabilities and software weaknesses in vehicles," in *Secure IT Systems* (Lecture Notes in Computer Science), A. Askarov, R. R. Hansen, and W. Rafnsson, Eds. Cham, Switzerland: Springer, 2019, pp. 204–218.

[29] A. Jøsang, M. Ødegaard, and E. Oftedal, "Cybersecurity through secure software development," in *Information Security Education Across the Curriculum* (IFIP Advances in Information and Communication Technology), M. Bishop, N. Miloslavskaya, and M. Theocharidou, Eds. Cham, Switzerland: Springer, 2015, pp. 53–63.

[30] A. Sadeghi, N. Esfahani, and S. Malek, "Mining the categorized software repositories to improve the analysis of security vulnerabilities," in *Fundamental Approaches to Software Engineering* (Lecture Notes in Computer Science), S. Gnesi and A. Rensink, Eds. Berlin, Germany: Springer, 2014, pp. 155–169.

[31] J. M. Kizza, "Introduction to computer network vulnerabilities," in *Guide to Computer Network Security* (Computer Communications and Networks), J. M. Kizza, Ed. Cham, Switzerland: Springer, 2020, pp. 87–103.

[32] S. Zhang, D. Caragea, and X. Ou, "An empirical study on using the national vulnerability database to predict software vulnerabilities," in *Database and Expert Systems Applications* (Lecture Notes in Computer Science), A. Hameurlain, S. W. Liddle, K.-D. Schewe, and X. Zhou, Eds. Berlin, Germany: Springer, 2011, pp. 217–231.

[33] Ú. Erlingsson, Y. Younan, and F. Piessens, "Low-level software security by example," in *Handbook of Information and Communication Security*, P. Stavroulakis and M. Stamp, Eds. Berlin, Germany: Springer, 2010, pp. 633–658.

[34] L. Ben Othmane, G. Chehrazi, E. Bodden, P. Tsalovski, and A. D. Brucker, "Time for addressing software security issues: Prediction models and impacting factors," *Data Sci. Eng.*, vol. 2, no. 2, pp. 107–124, Jun. 2017.

[35] F. Massacci, S. Neuhaus, and V. H. Nguyen, "After-life vulnerabilities: A study on Firefox evolution, its vulnerabilities, and fixes," in *Engineering Secure Software and Systems* (Lecture Notes in Computer Science), Ú. Erlingsson, R. Wieringa, and N. Zannone, Eds. Berlin, Germany: Springer, 2011, pp. 195–208.

[36] CWE. *Common Weakness Enumeration*. Accessed: Oct. 20, 2021. [Online]. Available: https://cwe.mitre.org/

[37] P. Rodeghero, S. Jiang, A. Armaly, and C. McMillan, "Detecting user story information in developer-client conversations to generate extractive summaries," in *Proc. IEEE/ACM 39th Int. Conf. Softw. Eng. (ICSE)*, May 2017, pp. 49–59.

[38] A. Ahmad, C. Feng, M. Khan, A. Khan, A. Ullah, S. Nazir, and A. Tahir, "A systematic literature review on using machine learning algorithms for software requirements identification on stack overflow," *Secur. Commun. Netw.*, vol. 2, pp. 107–124, Jun. 2017.

[39] A. Ahmad, "Research on comprehending software requirements on social media," Technol. Res., School Comput. Sci. Technol., Austria, Tech. Rep. 42, 2018.

[40] A. Ahmad, C. Feng, K. Li, S. M. Asim, and T. Sun, "Toward empirically investigating non-functional requirements of iOS developers on stack overflow," *IEEE Access*, vol. 8, pp. 501–506, 2019.
[41] M. Xiao, G. Yin, T. Wang, C. Yang, and M. Chen, "Requirement acquisition from social Q&A sites," in *Requirements Engineering in the Big Data Era*. Cham, Switzerland: Springer, 2015, pp. 64–74.

RAGHAVENDRA RAO ALTHAR received the bachelor's degree in mechanical engineering from the VTU and the M.B.A. degree in operations management from Indira Gandhi Open University. He is currently pursuing the Doctor of Philosophy degree in data science from CHRIST (Deemed to be University), Bengaluru, India. He is also working as a Quality Management Specialist with the Software Development Team of Insurance domain-based company. For last 15 years, he has been working on building quality management systems for various domains like manufacturing, retail, telecom and software industries, based on international standards like ISO, Six Sigma, and best global practices for industry. Adopting best practices of data science to optimize software development processes and connected business processes has been the recent focus area. He is a Six Sigma Black Belt Certified and a Quality Management & Information Security Audit Standards Certified Practitioner. His research interest includes application of data science in software development processes.

DEBABRATA SAMANTA (Member, IEEE) received the Ph.D. degree in computer science and engineering from the National Institute of Technology, Durgapur, India, in the area of SAR image processing. He is currently working as an Assistant Professor with the Department of Computer Science, CHRIST (Deemed to be University), Bengaluru, India, and he is also a Co-ordinator of the IPR Cell, since December 2019. He is keenly interested in interdisciplinary research and development and has experience spanning fields of SAR image analysis, video surveillance, heuristic algorithm for image classification, deep learning framework for detection and classification, blockchain, statistical modeling, wireless ad hoc networks, natural language processing, and V2I communication. He has successfully completed six consultancy projects. He has received funding 8,110 USD under Open Access, Publication Fund. He has received funding under the International Travel Support Scheme in 2019 for attending conference in Thailand. He has received the Travel Grant for Speaker in Conference, and Seminar, for two years, from July 2019. He is the owner of 21 patents (three design Indian patent and two Australian patent granted, and 16 Indian patents published) and two copyright. He has authored or coauthored over 197 research papers in international journals (SCI/SCIE/ESCI/Scopus) and conferences, including IEEE, Springer, and Elsevier Conference proceeding. He is the coauthor of 13 books and the co-editor of 11 books, available for sale on Amazon and Flipkart. He has presented various papers at international conferences and received best paper awards. He has authored or coauthored of eight book chapters. He also serves as an Acquisition Editor for Springer, Wiley, CRC, Scrivener Publishing LLC, Beverly, USA, and Elsevier. He is an Associate Life Member of Computer Society of India (CSI) and a Life Member of the Indian Society for Technical Education (ISTE). He has received "Scholastic Award" at the 2nd International Conference on Computer Science and IT Application, CSIT-2011, Delhi, India. He is a convener, keynote speaker, session chair, co-chair, publicity chair, publication chair, advisory board, and technical program committee members in many prestigious international and national conferences. He was an invited speaker at several institutions.

MANJIT KAUR (Senior Member, IEEE) received the Master of Engineering degree in information technology from Punjab University, Chandigarh, India, in 2011, and the Ph.D. degree from the Thapar Institute of Engineering and Technology, Patiala, India, in 2019. She worked as an Assistant Professor in three well-known universities of India, such as Chandigarh University, Mohali, India; Manipal University Jaipur, Jaipur, India; and Bennett University, Greater Noida, India. In 2021, she moved to the School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology, Gwangju, South Korea, where she is currently affiliated. She was in the top 2% list issues by "World Ranking of Top 2% Scientists," in 2021. She was part of the 14 Web of Science/Scopus indexed conferences. Her research interests include post-quantum cryptography, fully homomorphic encryption, and privacy-preserving machine learning, wireless sensor networks, digital image processing, and metaheuristic techniques.

DILBAG SINGH (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from the Thapar Institute of Engineering and Technology, Patiala, India, in 2019. He worked as an Assistant Professor in three well-known universities of India, such as Chandigarh University, Mohali, India; Manipal University Jaipur, Jaipur, India; and Bennett University, Greater Noida, India. In 2021, he moved to the School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology, Gwangju, South Korea, where he is currently affiliated. He was in the top 2% list issues by "World Ranking of Top 2% Scientists," in 2021. He was part of the 11 Web of Science/Scopus indexed conferences. He has published more than 80 research articles in SCI/SCIE indexed journals. He has also submitted five patents and has published three books and two book chapters. His H-index is 31. His research interests include image processing, computer vision, deep learning, metaheuristic techniques, and information security. He has acted as a Lead Guest Editor/an Editorial Board Member of many SCI/SCIE indexed journals, such as *Journal of Healthcare Engineering*, *Mathematical Problems in Engineering*, and *Journal of Intelligent and Fuzzy Systems*.

HEUNG-NO LEE (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of California at Los Angeles, Los Angeles, CA, USA, in 1993, 1994, and 1999, respectively. He was with HRL Laboratories, LLC, Malibu, CA, USA, as a Research Staff Member, from 1999 to 2002. From 2002 to 2008, he was an Assistant Professor with the University of Pittsburgh, Pittsburgh, PA, USA. In 2009, he moved to the School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology, Gwangju, South Korea, where he is currently affiliated. His research interests include information theory, signal processing theory, blockchain, communications/networking theory, and their application to wireless communications and networking, compressive sensing, future internet, and brain–computer interface. He has received several prestigious national awards, including the Top 100 National Research and Development Award in 2012, the Top 50 Achievements of Fundamental Research Award in 2013, and the Science/Engineer of the Month (January 2014).