A Study on Early & Non-Intrusive Security Assessment for Container Images

Mubin Ul Haque[2,3] and Muhammad Ali Babar[1,2,3]

[1] *Centre for Research on Engineering Software Technologies (CREST), University of Adelaide, Australia*

[2] *School of Computer Science, University of Adelaide, Australia*

[3] *Cyber Security Cooperative Research Centre, Australia*

Keywords: Container images, Configuration, Security, Non-Intrusive Assessment, Machine Learning

Abstract: The ubiquitous adoption of container images to virtualize the software contents bring significant attention in its security configuration due to intricate and evolving security issues. Early security assessment of container images can prevent and mitigate security attacks on containers, and enabling practitioners to realize the secured configuration. Using security tools, which operate in intrusive manner in the early assessment, raise critical concern in its applicability where the container image contents are considered as highly sensitive. Moreover, the sequential steps and manual intervention required for using the security tools negatively impact the development and deployment of container images. In this regard, we aim to empirically investigate the effectiveness of Open Container Initiative (OCI) properties with the Machine Learning (ML) models to assess the security without peeking inside the container images. We extracted OCI properties from 1,137 real-world container images and investigated six traditional ML models with different OCI properties to identify the optimal ML model and its generalizability. Our empirical results show that the ensemble ML models provide the optimal performance to assess the container image security when the model is built with all the OCI properties. Our empirical evidence will guide practitioners in the early security assessment of container images in non-intrusive way as well as reducing the manual intervention required for using security tools to assess the security of container images.

1 INTRODUCTION

Virtualization technology, e.g., containerization is gaining tremendous popularity among practitioners to develop and deploy their software infrastructures. For instance, Docker technology (Docker, 2021), which provides containerization solutions, has been ranked as the most popular technology for consecutive four years (2019-2022) in Stack Overflow (SoF) developers' survey (Overflow, 2022). The survey results indicate developers' massive interest in using container technologies. Besides, the SoF survey in 2022 revealed that the popularity of Docker has increased from 55% to 69% in 2022.

The reason of such popularity can be explained with the containerization capability of encapsulating the software contents, e.g., code, data, applications, and dependencies into a standalone executable unit, which is known as container images (Sultan et al., 2019; Pahl et al., 2017). Container images enable practitioners and system administrators to develop and deploy their containerized software in any operating system. Besides, container images are helpful to Continuous Integration and Deployment (CI/CD) practices which enable the rapid delivery of software by fulfilling ever-increasing business demand (Sultan et al., 2019) and requirements (Pahl et al., 2017).

Despite the benefits of utilizing container images, one of the major concerns is its security configuration, which is affected by the increased number of security issues (e.g., potential faults, security vulnerabilities, embedded malware) in container images (Haque and Babar, 2022; Javed and Toor, 2021; Wist et al., 2021). In a recent survey (RedHat, 2021) among practitioners and system administrators, ensuring the security configuration of container images has been mentioned as the top consideration (59%). Security professionals and researchers strongly advocated and recommended the early assessment of security issues in container images. Early assessment can not only prevent security attacks but also provide the confidentiality, integrity, and availability of the container images.

Static Analysis Tools (SAT), e.g., CLAIR (Clair, 2020), ANCHORE (Anchore, 2017), TRIVY (Trivy, 2020) play an essential role in the early assessment of security issues in container images (Javed and Toor,

Haque, M. and Babar, M.
A Study on Early Non-Intrusive Security Assessment for Container Images.
DOI: 10.5220/0011987900003464
In *Proceedings of the 18th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2023)*, pages 640-647
ISBN: 978-989-758-647-7; ISSN: 2184-4895

2021; Wist et al., 2021; Berkovich et al., 2020). However, using SAT to assess the security of container images raise several concerns. This assessment must need to access the container images, execute the tools, and then analyse the results, which is labour-intensive and require extensive domain expertise (Haque and Babar, 2022; Javed and Toor, 2021). Furthermore, earlier research efforts (Haque et al., 2020; Pahl et al., 2017) reported lack of domain experts in realizing the security of containerization technology since container development and deployment require expertise in various fields, such as software engineering, cloud computing, distributed networking and operating systems which are usually considered unnecessary in conventional software development and deployment.

Importantly, SAT operate in intrusive manner, e.g., they require to peek inside the container contents. This manner is conundrum with privacy since container owners run their container images on remote cloud servers and container contents should not be available to outside owners' purview (Cavalcanti et al., 2021). Container images have been ubiquitously adopted in mission-critical systems, healthcare, electric power systems, and banking systems, where container contents are considered as highly critical to access due to its sensitivity (Cui and Umphress, 2020). Besides, many government agencies are incorporating container images to provide critical services. For example, British Telecommunication incorporated containerization to develop and deploy their 5G use cases (Zhu and Gehrmann, 2021a). The highly sensitive contents in mission-critical and government owned container images stimulate a dire need of non-intrusive assessment of the security of the container images (Cui and Umphress, 2020), which can protect the privacy of the container as well as ensure the security configuration of such critical containers.

Previous research efforts (Zhu and Gehrmann, 2022; Zhu and Gehrmann, 2021a; Zhu and Gehrmann, 2021b; Cui and Umphress, 2020) for the security assessment of container images mainly performed the dynamic analysis, e.g., analysing the run-time entities of container images (e.g., system calls, capabilities, file access, resource usage). While this analysis required to go through several sequential and human intervened steps, such as building, installing, preparing the container images for execution, and then use a workload to monitor the run-time entities, there is a lack of research in assessing the container image security without performing such sequential as well as human intervened steps. This lack potentially hinders the release frequency of CI/CD practice, since the sequential steps of building, installing, preparing, and executing container images with a workload to monitor the run-time entities affecting the speed of container image development and deployment. In addition, large magnitude and highly frequent deployment of container images in CI/CD practice does not scale up with human intervention.

To assess the security of container images without building, installing, preparing, executing container images with a workload, and peeking inside the container images, we plan to study the Open Container Initiatives (OCI) properties of container images. This non-intrusive assessment will help practitioners and system administrators in the early security configuration in their development and deploying of virtualized software in cloud servers. The Open Container Initiative (OCI) is an open governance system for storing and distributing industry-standard container images (da Silva et al., 2018). We empirically investigate the effectiveness of learning models for the security assessment of container images leveraging the OCI properties.

In particular, our paper makes the following three main contributions:

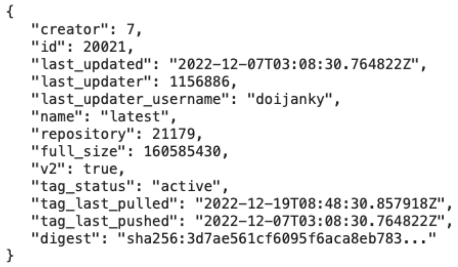
- To the best of our knowledge, this is the first automated support for the early and non-intrusive security assessment for container images leveraging OCI properties.
- We investigate the importance of OCI properties for the effectiveness of learning models for non-intrusive security assessment of container images.
- We investigate the generalizability of the learning models to assess the security of heterogeneous container image types.

2 BACKGROUND & RELATED WORK

In this Section, we briefly describe the Open Container Initiative properties and the prior studies that have investigated the non-intrusive security assessment of container images.

2.1 Open Container Initiative (OCI)

The acceptance of containers as a source of application storage, distribution, and portability necessitates the introduction of particular standards due to the rapid growth in both interest in and use of container-based virtualization. Due to the constant and immense expansion of Docker containerization, there is broad interest in a single and open container specification. This specification is not bound



```
{
   "creator": 7,
   "id": 20021,
   "last_updated": "2022-12-07T03:08:30.764822Z",
   "last_updater": 1156886,
   "last_updater_username": "doijanky",
   "name": "latest",
   "repository": 21179,
   "full_size": 160585430,
   "v2": true,
   "tag_status": "active",
   "tag_last_pulled": "2022-12-19T08:48:30.857918Z",
   "tag_last_pushed": "2022-12-07T03:08:30.764822Z",
   "digest": "sha256:3d7ae561cf6095f6aca8eb783..."
}
```

Figure 1: Example of OCI properties of *mysql:latest* image from Docker Hub.

to any particular container engine (e.g., Docker, rkt, CoreOS) or orchestration platform (e.g., Kubernetes, Nomad, Docker Swarm), or any commercial vendor or project, and portable across a wide range of operating systems, hardware, CPU architectures, and public clouds. In this regard, Open Container Initiative (OCI) was launched in 2015 by Docker along with other leading containerization service provider, such as CoreOS, to express the standardization for the purpose of creating open industry specification around container formats and runtime (Docker, 2021). Figure 1 shows an example of OCI properties for *mysql:latest* image from Docker Hub.

2.2 Non-Intrusive Security Assessment

Kwon and Lee proposed DIVDS (Docker Image Vulnerability Diagnostic System) to assess the security container images by analysing the encapsulated container contents (e.g., packages/libraries) and their vulnerability information with the help of a SAT, CLAIR (Kwon and Lee, 2020). They used a threshold score, defined by human expert to finalize the security assessment. Zerouali et al. proposed a technical lag framework to assess the security of Docker container images by analysing different lags, e.g., package, version, vulnerability (Zerouali et al., 2021). Brady et al. proposed a system to validate the security of container images by analysing the packages and their vulnerability information with the help of a SAT, ANCHORE (Brady et al., 2020). They also leveraged a manually defined threshold to finalize the security assessment.

Previous research typically followed the approach as (i) access encapsulated packages/libraries from container images, (ii) either use a SAT (e.g., ANCHORE/CLAIR) to identify vulnerable packages/libraries or use a vulnerability database (e.g., Ubuntu Security Tracker) to map packages/libraries with known vulnerability, (iii) a pre-defined threshold, generally provided by human, to finalize the security assessment. While this approach essentially relies on intrusive manner (e.g., required to access container image contents), however, we aim to assess the security in non-intrusive manner by investigating OCI properties, without analysing the container image contents.

3 RESEARCH SETTING

In this Section, we briefly describe our research questions, method, and data.

3.1 Research Questions (RQs)

Our study focuses on the following research questions.

- **RQ1.** What are the OCI properties that can be used to build the learning models for non-intrusive security assessment of container images? We aim to explore how well the Machine Learning (ML) models perform to learn the patterns derived from different OCI properties. An answer to this research question will help to understand the feasibility of OCI properties for learning-based model development to assess the security of container images in terms of non-intrusive manner. The answer will also enable us to identify the OCI properties which are providing the best predictive performance for ML models.
- **RQ2.** How effective is the learning models leveraging OCI properties to assess the security for cross container image types? We aim to explore how well the ML models perform to learn the OCI properties for cross container image types security assessment. Container images are developed (i.e., instantiated) on top of another container images and are deployed frequently in CI/CD practices where the container image types are highly diverse and heterogeneous (Haque et al., 2022; Sultan et al., 2019). An answer to this research question will help to understand whether the models trained on certain container iamge types can be generalized to perform prediction for another container image types. The answer will also enable us to inspect the generalizability and identify the best performing model to further utilize them in deployment phase of container images.

3.2 Research Method

The protocol for answering RQs is described here.

3.2.1 RQ1

We leveraged Mutual Information Gain (MIG) (Xu et al., 2007) technique to understand the importance of OCI properties for leveraging them in learning-based models to classify the containers from security point of view, whether the container is secured or insecured. MIG is a feature selection technique, which considers the joint probability of the features and their association with the target variables (Balogun et al., 2020; Xu et al., 2007) and used to identify important features for developing ML models to predict software defects in the existing literature (Balogun et al., 2020; Li et al., 2018; Wang et al., 2012). Besides, we designed the non-intrusive security assessment as a binary-class supervised classification problem from a learning perspective. To build ML models by using the OCI properties, we have considered all the OCI properties, which are last updated, name, tag last pulled, size, repository, tag status, digest, last updater username, creator, and last updater. The components for building a learning model include pre-processing, model selection, model building, and prediction.

Pre-processing is required since OCI properties contain noise (e.g., punctuation), which can make the learning model overfit (Luque et al., 2019; Kao and Poteet, 2007). Therefore, we used the state-of-the-art approaches (Sworna et al., 2022) for pre-processing the OCI properties, e.g., removal of noises and lowercasing. We used the pre-processed OCI properties to perform stratified k-fold cross-validation. Stratification ensures the ratio of each input source is kept throughout the cross-validation (Sworna et al., 2022), avoiding different data distribution of the folds.

Our model selection component has two steps as (i) feature engineering, (ii) model training and validation. Feature engineering is the process where OCI properties are transformed into features to improve the performance of the learning models. In the model training and validation steps, (k-1) folds were used for feature engineering and training a model, while the remaining one is used for validation. The validation performance of a model is the average of k runs. The model configurations with the highest performance metric would be selected as the optimal classifier for the following model building process.

The model building process used the pre-processed OCI properties to generate a feature model based on the identified feature configuration. The feature model has been saved to transform the data for future prediction. The prediction process is used for testing the trained model and classifying the container images for security assessment by leveraging the OCI properties. In this process, the OCI properties of a container image are first pre-processed and then transformed to a feature set using the saved feature model.

3.2.2 RQ2

For assessing the security of cross container image type, we chose one container image type as test set or target-container for prediction, while using other image types for training set or source-containers. In other words, we built a prediction model using the OCI properties and security labelling of source-containers, and predict the security labelling of target containers.

3.2.3 Evaluation Metric

We utilised the average Matthews Correction Coefficient (MCC) to evaluate the performance of ML models. MCC was used to select the optimal model since MCC explicitly considers all classes and is proclaimed as the best metric for error consideration by the prior study (Luque et al., 2019).

3.3 Research Data

We used the dataset provided by Haque et al. (Haque and Babar, 2022). This dataset contains the Docker container images labelled with security assessment in terms of pass and fail. In this dataset, a container image is labelled with pass if it does not contain any sort of security issues, otherwise, it is labelled as fail. They used a SAT, named ANCHORE and qualitatively investigated the assessment to create the dataset.

4 IMPLEMENTATION

Six traditional machine learning classifiers, Logistic Regression (LR), Naive Bayesian (NB), Support Vector Machines (SVM), Light Gradient Boosting Machine (LGBM), Decision Tree (DT) and Extreme Gradient Boosting (XGB), were selected for learning-based models. Those classifiers were chosen due to the common practice in the literature (Menzies et al., 2018; Ma et al., 2018). The first three (e.g., LR, NB, SVM) are single models, whereas the rest three (e.g., LGBM, DT, XGB) are ensemble models.

To select the optimal hyper parameter for each model, we performed stratified 10-fold cross-validation. Stratified sampling ensures that the proportion of each source would be kept. Moreover, one-tailed non-parametric Mann-Whitney U-test (Mann and Whitney, 1947) was calculated to compare the statistical significance of the observed samples. In our study, we considered 95% confidence with α (significance level) being 0.05, which is a statistical significance level (McKnight and Najab, 2010).

Table 1: Example of container type.

Type	Container Image Example
Analytics (AL)	logstash, piwik, telegraf
Database (DB)	mysql, redis, mongo,neo4j, postgres
Operating Systems (OS)	ubuntu, fedora, debian
Language Runtime (LR)	python, go, php
Base Images (BI)	alpine, bash, busybox
Messaging Services (MS)	nats, znc, rabbitmq, rocket.chat

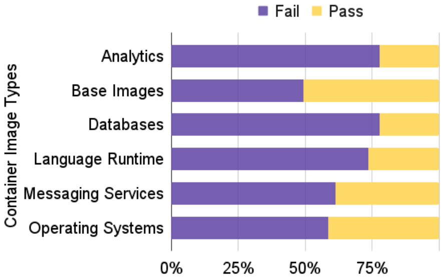


Figure 2: Distribution of data across container types.

To select optimal traditional ML models, we applied Bayesian optimization (Snoek et al., 2012) using hyperopt library (Bergstra et al., 2013). We chose bayesian optimisation due to its robustness against noisy objective function evaluations (Wang et al., 2013). We utilised the average Matthews Correction Coefficient (MCC) of 10-fold cross-validation with stratified sampling and early stopping criteria to select the optimal hyper parameters. Our dataset contains the security assessment labelling of 1,137 container images across six container types, which are Analytics (AL), Base Images (BI), Databases (DB), Language Runtime (LR), Messaging Services (MS), and Operating Systems (OS). The types of the containers had been collected from Docker Hub, and these types were also used in prior research studies (Kim et al., 2021). Table 1 shows some example container images for each of the types. Figure 2 shows the distribution of our dataset across container types, where 340 container images are labelled as pass or secured, and 797 container images are labelled as fail or insecured. Besides, Figure 3 shows the t-distributed Stochastic Neighbourhood Embedding (t-SNE) plot (Van der Maaten and Hinton, 2008) to visualize the structure of our high dimensional data in two dimensions.

5 EVALUATION RESULTS

The results of our RQs are described in this Section.

5.1 RQ1

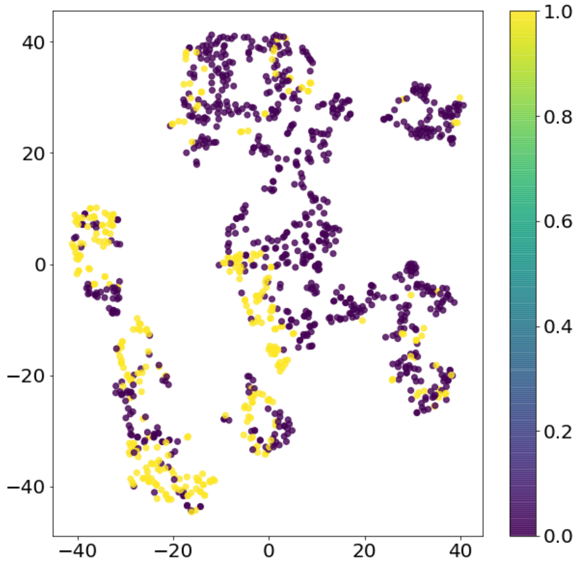


Figure 3: t-SNE plot for our dataset.

Table 2: Mutual Information Score for OCI properties.

# No	OCI Properties	Score
F1	Last Updated	0.339
F2	Name	0.294
F3	Tag Last Pulled	0.207
F4	Size	0.170
F5	Repository	0.156
F6	Tag Status	0.092
F7	Digest	0.053
F8	Last Updater Username	0.035
F9	Creator	0.021
F10	Last Updater	0.003

Table 2 demonstrates the mutual information score between each input variables, i.e, OCI properties with the output variable, i.e, security assessment. The higher score indicates close relationship between input variable and output variable (Balogun et al., 2020; Wang et al., 2012; Xu et al., 2007). It is evident from Table 2 that last updated time, image name, tag last pulled, size, and repository have higher (i.e., above average) score than rest of the OCI properties.

Table 3 shows the MCC values of six traditional and widely adopted ML models in practice while learning and predicting the security assessment of container images leveraging different OCI properties. It is observed from Table 3 that ensemble models,

Table 3: MCC for different ML models with different combination of OCI properties

OCI Properties	LR	NB	SVM	LGBM	DT	XGB
F1	0.486	0.499	0.480	0.671	0.645	0.643
F1 to F2	0.464	0.435	0.576	0.788	0.719	0.743
F1 to F3	0.420	0.478	0.546	0.785	0.736	0.746
F1 to F4	0.431	0.470	0.640	0.835	0.754	0.773
F1 to F5	0.447	0.425	0.607	0.854	0.794	0.793
F1 to F6	0.467	0.501	0.588	0.851	0.793	0.793
F1 to F7	0.459	0.501	0.600	0.846	0.770	0.798
F1 to F8	0.467	0.491	0.602	0.846	0.770	0.796
F1 to F9	0.457	0.495	0.587	0.850	0.773	0.796
F1 to F10	0.442	0.492	0.585	0.856	0.774	0.800

Table 4: Correlation between OCI properties and ML models

Co-efficient	LR	NB	SVM	LGBM	DT	XGB
ρ	-0.127	0.303	0.357	0.790	0.684	0.693
p-value	0.726	0.393	0.310	0.006	0.028	0.026

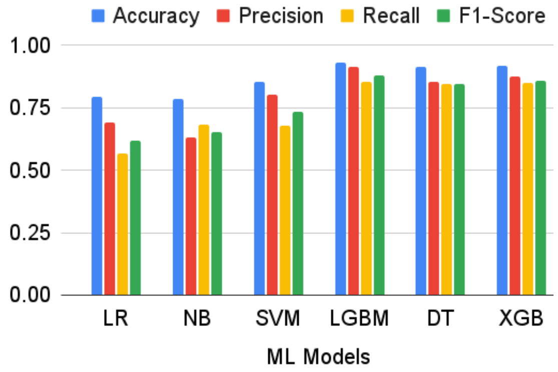


Figure 4: Evaluation results for the studied ML models.

such as, LGBM, DT, and XGB, perform better than the single models, such as, LR, NB, and SVM. The reason can be explained as the ensemble models use tree-based method to learn and aggregate the decisions to reduce the variance as well as maintain minimal bias (Ganaie et al., 2021). We verified our observation using the Mann-Whitney U-test (Mann and Whitney, 1947), where the z-score is -3.74185 and p-value is .00009, which is significant at $p < .05$.

Besides, we observed that the MCC values of ensemble models increase when the number of OCI properties increase. We verified our observation using Spearman correlation (ρ) test (Zar, 1972), where we found statistically significant and strong positive correlation between the number of OCI properties and the MCC values for ensembles models as shown in Table 4. We found LGBM and XGB with all ten properties (e.g., F1 to F10) and DT with five properties (e.g., F1 to F5) provide the optimal MCC score for the respective models.

Moreover, we observed that LGBM with all ten OCI properties (e.g., F1 to F10) achieves the best performance with respect to the all evaluation metrics among the studied ML models. We verified our observation using the Mann-Whitney U-test (Mann and Whitney, 1947), where the z-score is 2.68355 and p-value is .00368, which is significant at $p < .05$ with respect to DT and the z-score is 1.92762 and p-value is .0268, which is significant at $p < .05$ with respect to XGB. The MCC score of LGBM is 0.856 which is shown in Table 3, and accuracy is 0.932, precision is 0.915, recall is 0.853, and F1-score is 0.882, which are shown in Figure 4. LGBM performs better than the other models since it produces the trees in leaf-wise split which enables better learning of the input features (Ganaie et al., 2021). In summary, all of the OCI properties with ensemble models, in particular, LGBM, are effective to assess the security of container images.

Table 5: MCC of different ML models for cross container type security assessment.

Type	LR	NB	SVM	LGBM	DT	XGB
AL	0.492	0.492	0.336	0.659	0.579	0.524
BI	0.479	0.482	0.463	0.690	0.584	0.545
DB	0.320	0.332	0.286	0.358	0.348	0.358
LR	0.394	0.290	0.621	0.786	0.609	0.610
MS	0.953	0.904	0.594	0.953	0.859	0.772
OS	0.304	0.271	0.304	0.453	0.310	0.414

5.2 RQ2

Table 5 demonstrates the MCC score of the six ML models while learning from the source container types and predicting the security assessment of containers of another container type by leveraging OCI properties. It is observed from the Table 5 that ensemble models perform better than the single models in the evaluation metrics except Messaging Services (MS) and Databases (DB) container types. LR performs similar to LGBM while assessing the security for Messaging Services (MS) container types. The reason can be explained with the fact of very small number of testing set, as we had only 32 testing data.

Besides, we identified low MCC score (e.g., below 0.4) in the container types where the training dataset is much smaller, indicating difficulties for the models to learn OCI patterns for cross container security assessment. For example, in Database (DB) category, we identified the maximum MCC score 0.358 for LGBM and XGB, as there were only 585 training data. On the other hand, we identified LGBM performs better than the other ML models while assessing the cross container type security as shown in Table 5. In summary, ensemble models, in particular, LGBM, are effective to leverage the OCI properties to assess the security of cross container image types.

6 IMPLICATION

Implications for Practitioners. Our empirical results will benefit the developers and system administrators while securing the configuration of container images in non-intrusive manner. Developers can adopt the best performing ML models, for example, LGBM, to generate the secure candidate pool of container images from hundred thousands of container images without accessing its internal contents. In addition, our empirical result shows that the ML models do not require any kind workload to assess the security of container images. Earlier research studies have discussed the lack of container workload in containerized context, which is negatively impacting system administrators in deploying the container images in servers (Kim et al., 2021; Cavalcanti et al., 2021; Cui and Umphress, 2020). In this regard, our research provides a novel and effective solution to assess the security without using any kind of workloads. Besides, leveraging OCI properties to develop and utilize the learning-based model to assess the container images security can significantly reduce the manual intervention steps (e.g., building, preparing, and executing the container). In addition, our empirical result shows prominent predictive performance for cross container type security assessment by using OCI properties, which can encourage developers and system administrators to build or develop their own ML models even though there is no training data for a particular container type.

Implications for Researchers. Our empirical results shows a novel approach to assess the container image security using OCI properties and ML models. Researchers can further investigate how the Deep Learning (DL) models, such as, Convolutional Neural Network (CNN), perform to assess the container image security using OCI properties, where our empirical results for ML models can be used a baseline. In addition, we demonstrated that lower number of training samples can result in poor performance for ML models while assessing the security of cross container types. Future research can investigate text data augmentation (Sworna et al., 2022) to increase the training data samples and its performance for security assessment. Besides, our novel approach will benefit researchers to further investigate how OCI properties can be represented to the ML models for severity assessment of the security vulnerabilities of the container images.

7 CONCLUSION

Practitioners' preference of developing and deploying virtualized software has observed an exponential growth due to the encapsulation of application, code, data and dependencies in the form of container images. This encapsulation helps to reuse and share the software component and thus enabling practitioners to overcome one of the key challenge, timely delivery of the software. Intrusive way of operating the security tools in the early security assessment of container images bring crucial challenges in terms of its usage in highly sensitive containers. In addition, the sequential steps and manual intervention required for operating the security tools obstruct the rapid container image development and delivery. Our empirical evidence demonstrates a novel approach for non-intrusive security assessment of the container images by leveraging the OCI properties and ML models. We showed that the ensemble ML model, for example, LGBM, achieves the best predictive performance than the other ML models for non-intrusive security assessment when trained with all the OCI properties. In our future work, we will investigate the effectiveness and importance of OCI properties in the deep learning-based models for non-intrusive security assessment.

ACKNOWLEDGEMENT

The work has been supported by the Cyber Security Research Centre Limited whose activities are partially funded by the Australian Government's Cooperative Research Centres Programme.

REFERENCES

Anchore (2017). Anchore. https://anchore.com. Access Date Jan, 2023.

Balogun, A. O., Basri, S., Mahamad, S., Abdulkadir, S. J., Almomani, M. A., Adeyemo, V. E., Al-Tashi, Q., Mojeed, H. A., Imam, A. A., and Bajeh, A. O. (2020). Impact of feature selection methods on the predictive performance of software defect prediction models: an extensive empirical study. *Symmetry*, 12(7):1147.

Bergstra, J., Yamins, D., and Cox, D. (2013). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International conference on machine learning*, pages 115–123. PMLR.

Berkovich, S., Kam, J., and Wurster, G. (2020). {UBCIS}: Ultimate benchmark for container image scanning. In *13th USENIX Workshop on Cyber Security Experimentation and Test (CSET 20)*.

Brady, K., Moon, S., Nguyen, T., and Coffman, J. (2020). Docker container security in cloud computing. In *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0975–0980. IEEE.

Cavalcanti, M., Inacio, P., and Freire, M. (2021). Performance evaluation of container-level anomaly-based intrusion detection systems for multi-tenant applications using machine learning algorithms. In *The 16th International Conference on Availability, Reliability and Security*, pages 1–9.

Clair (2020). Clair. https://github.com/quay/clair. Access Date Jan, 2023.

Cui, P. and Umphress, D. (2020). Towards unsupervised introspection of containerized application. In *2020 the 10th International Conference on Communication and Network Security*, pages 42–51.

da Silva, V. G., Kirikova, M., and Alksnis, G. (2018). Containers for virtualization: An overview. *Appl. Comput. Syst.*, 23(1):21–27.

Docker (2021). Docker documentation. https://docs.docker.com. Access Date August, 2022.

Ganaie, M., Hu, M., et al. (2021). Ensemble deep learning: A review. *arXiv preprint arXiv:2104.02395*.

Haque, M. U. and Babar, M. A. (2022). Well begun is half done: An empirical study of exploitability & impact of base-image vulnerabilities. In *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 1066–1077. IEEE.

Haque, M. U., Iwaya, L. H., and Babar, M. A. (2020). Challenges in docker development: A large-scale study using stack overflow. In *Proceedings of the 14th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–11.

Haque, M. U., Kholoosi, M. M., and Babar, M. A. (2022). Kgsecconfig: A knowledge graph based approach for secured container orchestrator configuration. In *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 420–431. IEEE.

Javed, O. and Toor, S. (2021). An evaluation of container security vulnerability detection tools. In *2021 5th International Conference on Cloud and Big Data Computing (ICCBDC)*, pages 95–101.

Kao, A. and Poteet, S. R. (2007). *Natural language processing and text mining*. Springer Science & Business Media.

Kim, S., Kim, B. J., and Lee, D. H. (2021). Prof-gen: Practical study on system call whitelist generation for container attack surface reduction. In *2021 IEEE 14th International Conference on Cloud Computing (CLOUD)*, pages 278–287. IEEE.

Kwon, S. and Lee, J.-H. (2020). Divds: Docker image vulnerability diagnostic system. *IEEE Access*, 8:42666–42673.

Li, Z., Jing, X.-Y., and Zhu, X. (2018). Progress on approaches to software defect prediction. *Iet Software*, 12(3):161–175.

Luque, A., Carrasco, A., Martín, A., and de las Heras, A. (2019). The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, 91:216–231.

Ma, Y., Fakhoury, S., Christensen, M., Arnaoudova, V., Zogaan, W., and Mirakhorli, M. (2018). Automatic classification of software artifacts in open-source applications. In *2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR)*, pages 414–425. IEEE.

Mann, H. B. and Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60.

McKnight, P. E. and Najab, J. (2010). Mann-whitney u test. *The Corsini encyclopedia of psychology*, pages 1–1.

Menzies, T., Majumder, S., Balaji, N., Brey, K., and Fu, W. (2018). 500+ times faster than deep learning:(a case study exploring faster methods for text mining stackoverflow). In *2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR)*, pages 554–563. IEEE.

Overflow, S. (2022). Stack overflow survey results. https://insights.stackoverflow.com/survey. Jan, 2023.

Pahl, C., Brogi, A., Soldani, J., and Jamshidi, P. (2017). Cloud container technologies: a state-of-the-art review. *IEEE Transactions on Cloud Computing*.

RedHat (2021). State of kubernetes security report. https://www.redhat.com/en/engage/state-kubernetes-security-s-202106210910. Access Date August, 2022.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Sultan, S., Ahmad, I., and Dimitriou, T. (2019). Container security: Issues, challenges, and the road ahead. *IEEE Access*, 7:52976–52996.

Sworna, Z. T., Islam, C., and Babar, M. A. (2022). Apiro: A framework for automated security tools api recommendation. *ACM Transactions on Software Engineering and Methodology*.

Trivy (2020). Trivy. https://github.com/aquasecurity/trivy. Access Date Jan, 2023.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Wang, P., Jin, C., and Jin, S.-W. (2012). Software defect prediction scheme based on feature selection. In *2012 Fourth International Symposium on Information Science and Engineering*, pages 477–480. IEEE.

Wang, Z., Zoghi, M., Hutter, F., Matheson, D., and De Freitas, N. (2013). Bayesian optimization in high dimensions via random embeddings. In *Twenty-Third international joint conference on artificial intelligence*.

Wist, K., Helsem, M., and Gligoroski, D. (2021). Vulnerability analysis of 2500 docker hub images. In *Advances in Security, Networks, and Internet of Things*, pages 307–327. Springer.

Xu, Y., Jones, G. J., Li, J., Wang, B., and Sun, C. (2007). A study on mutual information-based feature selection for text categorization. *Journal of Computational Information Systems*, 3(3):1007–1012.

Zar, J. H. (1972). Significance testing of the spearman rank correlation coefficient. *Journal of the American Statistical Association*, 67(339):578–580.

Zerouali, A., Mens, T., Decan, A., Gonzalez-Barahona, J., and Robles, G. (2021). A multi-dimensional analysis of technical lag in debian-based docker images. *Empirical Software Engineering*, 26(2):1–45.

Zhu, H. and Gehrmann, C. (2021a). Apparmor profile generator as a cloud service. In *CLOSER*, pages 45–55.

Zhu, H. and Gehrmann, C. (2021b). Lic-sec: an enhanced apparmor docker security profile generator. *Journal of Information Security and Applications*, 61:102924.

Zhu, H. and Gehrmann, C. (2022). Kub-sec, an automatic kubernetes cluster apparmor profile generation engine. In *2022 14th International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pages 129–137. IEEE.