ITU Journal of Wireless Communications and Cybersecurity

ITU/JWCC

# Ransomware, Spyware, and Trojan Malware Detection for Android Using Machine Learning

**Swati Shilaskar[1], Shripad Bhatlawande[1], Akhil Bhalgat[1, 2], and Niranjan Bharate[1, 3]**

[1] Department of Electronics and Telecommunication Engineering Vishwakarma Institute of Technology, Pune, 411037, India
[2] School of Management, University of Texas, Dallas, 75080, USA
[3] Devtech Consulting, Pune, 411015, India

**Abstract:** The threat posed by malware has increased with the growth of technology. This makes malware detection a crucial problem. It specifically pertains to the heightened security risks that the underlying programs and their users frequently encounter. On the CIC-MalMem2022 dataset, experiments were executed. KNN, Decision Tree, Random Forest, GaussianNB, and AdaBoost were used for binary classification and multiclass classification. Additionally, the effectiveness of the employed algorithms has been evaluated. The machine learning models were optimized by tuning the hyperparameters. Random Forest and AdaBoost both achieved binary classification accuracy of 99.99%. Optuna Hyperparameter tuning for Random forest based multiclass classification performed with an accuracy of 88.31%.

**Keywords:** Android, malware detection, CICMalmem-22, machine learning, Optuna.

## Makine Öğrenimini Kullanarak Android için Fidye Yazılımı, Casus Yazılım ve Truva Atı Kötü Amaçlı Yazılım Tespiti

**Özet:** Teknolojinin gelişmesiyle birlikte kötü amaçlı yazılımların oluşturduğu tehdidin de artış göstermesi kötü amaçlı yazılım tespitini önemli bir sorun haline getirmektedir. Bu da özellikle temel programların ve kullanıcılarının sıklıkla karşılaştığı yüksek güvenlik riskleriyle ilgilidir. CIC-MalMem2022 veri setinde deneyler gerçekleştirildi. İkili sınıflandırma ve çok sınıflı sınıflandırma için KNN, Karar Ağacı, Rastgele Orman, GaussianNB ve AdaBoost kullanıldı. Ayrıca kullanılan algoritmaların etkinliği de değerlendirilmiştir. Makine öğrenimi modelleri, hyperparametreler ayarlanarak optimize edildi. Random Forest ve AdaBoost'un her ikisi de %99,99'luk ikili sınıflandırma doğruluğuna ulaştı. Rastgele orman tabanlı çok sınıflı sınıflandırma için Optuna Hiperparametre ayarı %88,31 doğrulukla gerçekleştirildi.

**Anahtar Kelimeler:** Android, kötü amaçlı yazılım tespiti, CICMalmem-22, makine öğrenme, Optuna.

RESEARCH PAPER

Corresponding Author: Swati Shilaskar, swati.shilaskar@vit.edu

Reference: S. Shilaskar, S. Bhatlawande, A. Bhalgat, and N. Bharate, (2024), "Ransomware, Spyware, and Trojan Malware Detection for Android Using Machine Learning," *ITU Journ. Wireless Comm. Cyber.*, 1, (1) 1–8.

Submission Date: Feb, 14, 2024
Acceptance Date: May, 17, 2024
Online Publishing: Sept, 30, 2024

## 1 INTRODUCTION

The term "malware," which is derived from "malicious software," describes any program designed to damage computer systems or steal confidential information. One of the biggest threats to computer security is malware since it may hurt both individuals and businesses. Malware becomes increasingly sophisticated as technology develops, which makes it more challenging to identify and avoid. As a result, efficient malware detection systems are now more crucial than ever. With a global market share of 71.64%, Android phones are predicted to be owned by 3.3 billion people globally, making them the most frequently used operating system [1]. Therefore, this makes Android a prime target for malware attacks. It greatly jeopardizes users' security and privacy. End-to-end technique for analyzing characteristics extracted from Android applications is described in [2] and is based on Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNNs). It automatically connects static and dynamic features to assess if a program is harmful. They had a 96.76% accuracy rate. To report current problems faced while detecting malware, [3] used the visualization of data and adversarial learning on ML-based classifiers to efficiently find various malwares groups, taking into account threats from adversarial instances and the enormous rise in malware variants. They addressed a deep feature extraction approach to analyze malware in the context of recent deep learning (DL) advancements. These features from CNN were obtained and entered into Support Vector Machine (SVM) classifier in order to categorize malware [4]. By showcasing the inadequacies and constraints of the existing malware analysis tools, Severi et al. [5] developed a new system. They wanted to gather the system's full set of traces and enable replication using the Malrec platform. Petrik et al. [5] [6] carried out Simple binary classification from raw data and device memory dumps were used to identify malware. The results indicated that it is not accurate enough to determine the various characteristics of stride length and time. The usefulness of DL algorithms in the classification of malware was examined [7]. This study evaluated the malware detection algorithms Long short-term memory (LSTM), Gated Recurrent Unit (GRU), and CNN for static and behavior-based detection. Concurrently, a combination of CNN and LSTM models was developed. With 99.31% accuracy, the suggested hybrid model surpassed the competition. In addition, a random forest classifier was created by Ahmadi, Wüchner and Mao et al. [8] [9] ([10]) to identify malware using a number of factors, namely system calls, the file system, and other features. A DL-based approach for automated Android malware classification is presented by McLaughlin et al. [11]. The solution integrates deep neural networks and static feature analysis to extract properties from the code and Android application's metadata. CNN and LSTM networks are combined to form deep neural network's architecture of the system. The system achieved 99% accuracy on a dataset of 10,000 Android applications. The study highlights the importance of feature selection and engineering in achieving high detection accuracy and demonstrates the promise of DL models for automatic detection of malware. The approach used by Vinayakumar, et al. [12] for extraction of features from the code and metadata of Android applications combines static analysis and LSTM networks. To understand malware patterns and characteristics, the LSTM network is trained on a vast sample of harmless and malicious Android applications. On the dataset of 3,824 Android applications, the system obtained an accuracy of 98.12%. The study shows that LSTM networks are successful in detecting Android malware and emphasizes the significance of feature selection and engineering in achieving high detection accuracy. The suggested approach has the potential to identify previously undiscovered and unknown malware. Analysis of the efficacy of a single extracted feature from APK files for binary classification in malware detection was performed . It was examined on a dataset of 4,992 Android applications and using the SVM, attaining a 95.1% accuracy [13] . The suggested approach has limits in identifying increasingly complicated and sophisticated malware, which may necessitate additional characteristics. In order to find patterns in system-wide data, namely the storage available and transferred packet consumption, the authors proposed a customized DL system based on prevailing models like Encoder and the ResNet model [14]. This system would be used to detect sensitive app behaviors. using a machine learning-based approach to malware classification for Android called Random Forest classification [15]. To extract features from the code and metadata of Android applications, the proposed approach utilizes a hybrid approach of static analysis and feature engineering. Carrier, T. et al. [16] propose an approach to extract properties from the memory of Android applications that requires both static and dynamic features. Then, machine learning classifier is developed using the information gathered to distinguish between benign and adverse applications, including those that have been hidden. The system's accuracy on the CiC-Malmem -22 dataset was 99.7% .

## 2 METHODOLOGY

The proposed method uses machine learning classifiers like Random Forest, KNN and Decision Tress to detect malicious applications. The proposed technique also classifies the malignant applications into Benign, Ransomware, Spyware and Trojans. The short flowchart of the proposed method is shown in Fig. 1.

The proposed method leverages the CiCMalmem-2022 dataset, designed specifically to address the challenge of obfuscated malware detection. Obfuscated malware refers to malicious software deliberately concealed to evade detection and removal. This dataset was meticulously cu-

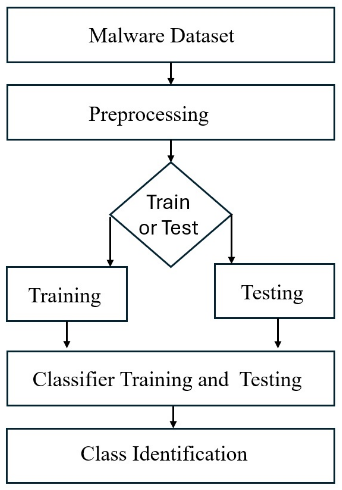


**Fig. 1** Process flow of the system.

rated to emulate real-world scenarios, featuring prevalent strains of obfuscated malware encountered in actual cyber threats. This dataset comprises a total of 58,569 records. The collection of harmful and benign dumps yields, 29,298 benign and 29,298 malicious entries. Within the malicious category, the samples are further classified into three main classes: ransomware, spyware, and Trojan horses, with 9,535, 9,866, and 9,907 entries respectively. It's worth noting that while ransomware, spyware, and Trojan horses are subdivided into subclasses, these subclasses were not utilized in the current study. The dataset comprises several distinct feature sets, each offering insights into different aspects of malware behavior. These feature sets include pslist, dlllist, handles, ldrmodules, malfind, psxview, modules, and svcscan. pslist, which provides details on process listing, such as processes, parent processes, and average thread count per process, encompassing 5 features; dlllist, which encompasses details about DLL (Dynamic Link Library) usage, covering DLLs loaded by processes and the average number of DLLs per process, with 2 features; handles, offering insights into handle usage, providing information on the total number of handles and the average number of handles per process, with 9 features; ldrmodules, capturing details about loaded modules, including modules not loaded, initialized, and in memory, comprising 6 features; malfind, focused on memory scanning for malware, detecting memory injections, with 3 features; psxview, providing an overview of process views, including processes not listed, not in specific process pools, and not in specific lists, consisting of 14 features; modules, offering insights into loaded modules, with 1 feature; and svcscan, providing information on scanned services, kernel drivers, and process services, consisting of 7 features. These feature sets, each with its respective number of features, collectively contribute to the comprehensive analysis of malware behavior in the dataset. 26 Features extracted by using listed VolMemLyzer feature extractor are listed in [17]. The accuracy of 99.99% is achieved for binary classification in the pioneering work described in this paper.

For the dataset to be appropriate for categorization, some data preparation procedures are needed. These techniques are crucial for increasing the potency of classification models and transforming data into a machine-learnable format. In this study, the categorical variables In order to prepare them for machine learning classification, Benign and Malicious are given the two different values of 0 and 1, respectively. The CIC-MalMem2022 dataset, on the other hand, is a dataset that is evenly balanced with two classes: benign and malware. It shows resilience to overfitting since the dataset is evenly balanced. Additionally, we removed elements from the dataset that don't affect machine learning's effectiveness. These characteristics have zero weights and have no bearing on how the learning algorithms perform. In this research, The k-fold validation method is employed here. With this technique, the data is at random segregated into two groups: training and testing set. Then, this dataset is divided into 'k' samples, with 'k-1' samples being utilized for training and 'k' samples for testing . This entire procedure is performed k times with different training and testing datasets each time. The optimal model is then chosen based on the lowest error produced via the use of several statistical methods for error estimates. Random Forest is a supervised classification and regression ensemble learning system [18] . It entails building numerous decision trees that produce predictions using a randomly selected subset of features. During training, the technique includes randomization to reduce overfitting and make the model more resilient [19]. During prediction, the algorithm takes the input data and sends it through each decision tree, with the final prediction determined by the majority vote of all the trees [20]. Random forest is a versatile and strong algorithm that may offer a measure of feature relevance and can be utilized for a variety of purposes. A common machine learning method for classification and regression issues is decision trees. Creating a tree-like model of decisions and probable outcomes is one of them. The algorithm looks for the ideal properties to divide the data into subsets and produce decision nodes that maximize information gain or minimize impurity during training. The model can be visualized as an if-then flowchart, with every internal node showing a test on a feature, each branch denoting the test's output, and every leaf node denoting an outcome or prediction [21]. Decision trees can handle both category and numerical data and are easily interpretable. They are, however, susceptible to overfitting and can be sensitive to slight changes in the training data. AdaBoost, or Adaptive Boosting, is an ensemble learning technique that combines numerous weak classifiers to produce a powerful classifier.

It accomplishes this by iteratively training weak classifiers on different subsets of the training data, with a focus on examples that were misclassified in prior iterations. The information and experience of these weak classifiers are aggregated to generate the final classifier. AdaBoost successfully identifies complicated decision limits that precisely represent the underlying structure of the data via this iterative boosting method. The technique for regression problems locates the K data points in the training set that are closest to the input data point and assigns the most frequent class or the mean value of the K closest neighbors to those data points. Any distance metric, including Manhattan distance and Euclidean distance, can be used to determine how similar two data points are. It also offers precise end-user predictions established on the basic categorization principles of resemblance or distance [22]. The Gaussian Naive Bayes method is a probabilistic algorithm that computes the probability of various classes given input characteristics. It entails conducting calculations like median, variance, and probabilistic density projections, which can be computationally demanding, particularly when dealing with big datasets or high-dimensional feature spaces. Grid Search: Finding the set of hyperparameters that results in the best model performance necessitates constructing a grid of hyperparameter values and carefully examining all potential combinations of these values. In order to perform a grid search, we might need to specify a boundary because the parameter space for the machine learning approach could comprise spaces with actual or infinite values for some parameters. The hyperparameters that yield the best results are often chosen after the model's performance has been evaluated using a performance metric like accuracy, precision, or recall. The implementation is shown with a flowchart in Fig. 2.

Optuna Hyperparameter Tuning: It employs a rapid algorithm based on Bayesian optimization to search the hyperparameter space and discover the best set of hyperparameters for a given model. The framework operates by defining the area of search space for the hyper parameters and the optimization target in an experiment object. Then, with the purpose of minimizing the objective function, it employs a mix of Tree structured Parzen estimators (TPE) and other optimization algorithms to intelligently sample the hyperparameters and assess their performance [23]. This is shown in Fig. 3.

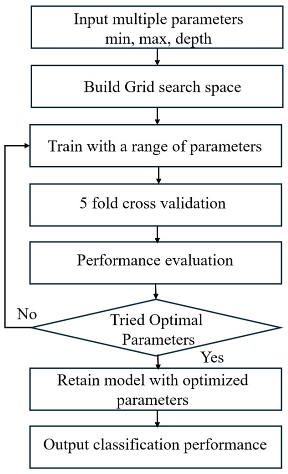

Input multiple parameters min, max, depth
Build Grid search space
Train with a range of parameters
5 fold cross validation
Performance evaluation
Tried Optimal Parameters
No
Yes
Retain model with optimized parameters
Output classification performance

**Fig. 2** Flowchart of Grid Search.

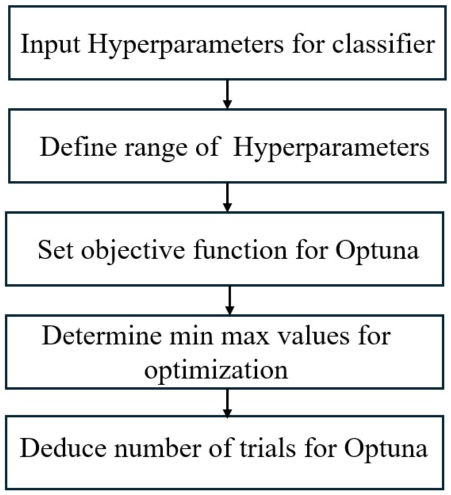

Input Hyperparameters for classifier
Define range of Hyperparameters
Set objective function for Optuna
Determine min max values for optimization
Deduce number of trials for Optuna

**Fig. 3** Selection of Optuna hyperparameters.

Algorithm 1 describes hyperparamaeter selection process for Optuna optimization.

**Algorithm 1** Optuna Hyperparameter Algorithm

```
1. Input dataset
2. for metric 1 to 100 do
3. Splitting dataset into 70% training set and 30% test set
4. Fit Min_Max_Scaler to training set.
5. for i in n_trials do
6. select new a_(n+1) by optimizing function
7. a_(n1)ārgmax Alpha(a;R_n )
8. query objective function to get b_(n+1)
9. amplify data R_(n1)
10. Compute metric for best model over test set
11. if run-time > time_limit then
12. end
```

## 2.1 Model Implementation and Evaluation

In this section, the proposed system uses an ensemble approach for binary classification of malware, into malignant and benign. The six classifiers that were used in the approach are mentioned in the Table 1.

**Table 1** Accuracy for binary classification of malware

| Classifier | Accuracy in% |
|---|---|
| Decision Tree | 99.96 ± 0.02 |
| KNN | 99.92 ± 0.21 |
| Random Forest | 99.99 ± 0.005 |
| Gaussian NB | 99.25 ± 0.06 |
| AdaBoost | 99.99 ± 0.05 |
| Logistic Regression | 99.57 ± 0.17 |

To further classify the malignant files into their respective malware families, we also performed a multiclass classification of the CiCMalmem-22 dataset. In order to improve the parameters of the base learner and get the highest classification accuracy for Android malware applications, Hyperparameter tuning techniques such as, Grid search and Optuna were used. The correlation matrix heatmap is a graphical representation of data that is used to identify the strongest correlations between various characteristics and the target feature. Each feature in a dataset is represented by a different colour, which informs about the relationships between the features. For the demonstration, we created a heatmap of the correlation matrix, Fig. 4. As given in the Fig. 4 numbers are given to the feature names and are listed in next paragraph.

Selection of the features is then carried out to remove redundant and unnecessary features. This helps in dimensionality reduction of the data and make it more efficient. Only features with correlation value of 0.95 and above were eliminated, and the rest of were considered. Total 55 features are present in the dataset, out of which 3 are eliminated due to non variability. 52 features used to train the classifiers are given as follows: 1. pslist.nproc, 2. nppid, 3. avg_threads, 4. handlers 5. dlllist.ndlls, 6.avg_dlls_per_proc, 7. handles.nhandles, 8. avg_handles_per_proc, 9. nfile, 10. nevent, 11. ndesktop, 12. nkey, 13. nthread, 14. ndirectory, 15. nsemaphore, 16. ntimer, 17. nsection, 18. nmutant; 19. ldrmodules.not_in_load, 20. init, 21. mem, 22. load_avg, 23. init_avg, 24. mem_avg 25. malfind.ninjections, 26. commitCharge, 27. protection, 28. uniqueInjections 29. psxview.not_in_pslist, 30. eprocess_pool, 31. ethread_pool, 32. pspcid_list, 33. csrss_handles, 34. session, 35. deskthrd, 36. pslist_false_avg, 37. eprocess_pool_false_avg, 38. ethread_pool_false_avg, 39. not_in_pspcid_list_false_avg, 40. csrss_handles_false_avg, 41. session_false_avg, 42. deskthrd_false_avg 43. modules.nmodules, 44. svcscan.nservices, 45. kernel_drivers, 46. fs_drivers, 47. process_services, 48. shared_process_services, 49. nactive; 50. callbacks.ncallbacks, 51. nanonymous, 52. ngeneric. Only 32 features are selected as shown in Fig. 5. The selected feature as listed in Fig. 5 are pslist.nproc, nppid, avg_threads, avg_handlers; dlllist.ndlls, avg_dlls_per_proc; handles.ndesktop, nkey, nsemaphore, nsection, nmutant; ldrmodules.not_in_load, load_avg, malfind.ninjections, commitCharge, uniqueInjections, modules.nmodules, psxview.not_in_pslist, eprocess_pool, ethread_pool, eprocess_pool, false_avg, ethread_pool, false_avg; svcscan.nservices, kernel_drivers, fs_drivers, process_services, nactive; callbacks.ncallbacks, nanonymous, generic.

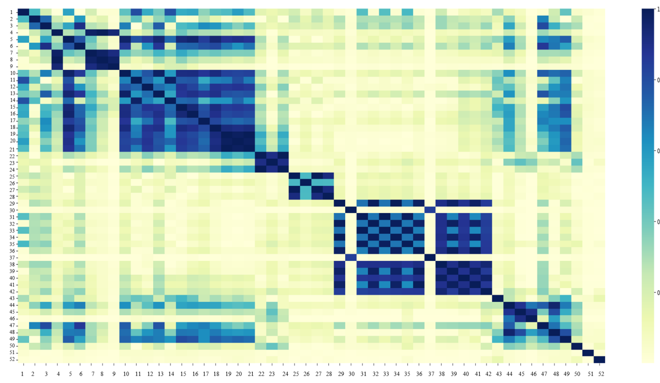


**Fig. 4** Correlation heatmap of all features.

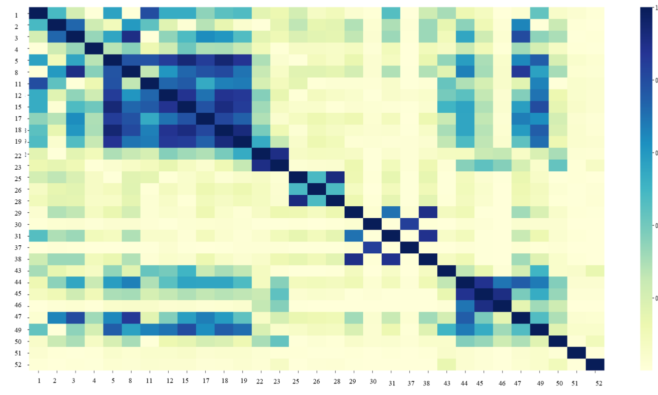


**Fig. 5** Correlation heatmap of the reduced features.

In the suggested learning technique, the Random Forest algorithm was first run with the default parameters, to gauge the level of its accuracy. To further optimize the algorithm, Grid search method was employed. Fig. 6 shows the comparison of how depth of the tree impacts accuracy for Random Forest with 20, 60, and 130 trees.

As can be seen from the figure, stable accuracy is achieved at 23 depths of tree. We notice similar trends for other tree sizes. When the amount of trees in the Random

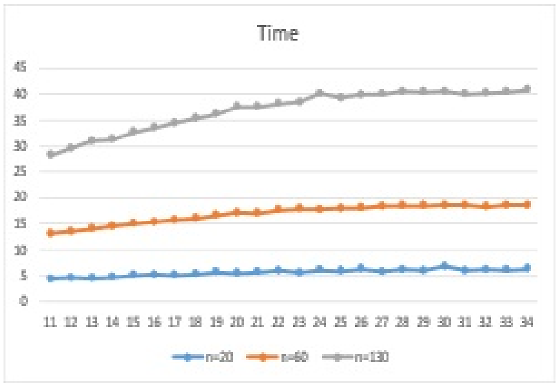


**Fig. 6** Time for trees and maximum allowed depth.

Forest increases, so does its computational time. Overall trend is depicted in Fig. 7

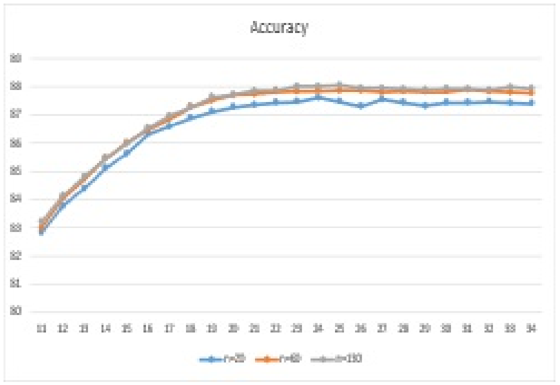


**Fig. 7** Accuracy comparison with 20, 60 and 130 trees for varying depth of tree.

To optimize decision trees, varying depths were evaluated. The maximum allowed depth was 10 through 50, and their accuracy criterion was evaluated. The depth that gives the best performance was chosen, as shown in Table 2. The maximum accuracy is achieved at the depth tree of 23 with accuracy of 85.05%. It becomes stagnant as soon as the depth of trees is increased; however, the computational time also increases substantially.

By comparing the weights of numerous existing features, a case search method known as nearest neighbor determines how similar two instances are. The number of neighbors in a group of training data which lie closest to a specific value in the validation or testing set is represented by the K parameter of K-NN classifiers. K-values between 2 and 40 were examined. In Fig. 8, accuracy as K-values changed is observed.

Accuracy continues to decline as the K value of K - Nearest Neighbors increases. To improve the classification performance more samples are included to train the GaussianNB model. The datapoints that are away from the distribution mean are taken in consideration for building the model. The parameter is named var_smoothing for GaussianNB. It is plotted on x axis and accuracy on y axis to show the performance, as seen in Fig. 9.

**Table 2** Accuracy mean and standard deviation performance of decision tree for varying depths

| Depth | Time (s) | Accuracy (mean ± std) in % |
|---|---|---|
| 10 | 1.6 | 80.81 ± 0.63 |
| 11 | 2.32 | 81.45 ± 0.73 |
| 12 | 1.83 | 82.22± 0.39 |
| 13 | 1.95 | 82.97± 0.20 |
| 14 | 1.88 | 83.51 ± 0.34 |
| 15 | 2.3 | 83.94 ± 0.31 |
| 16 | 2.1 | 84.18 ± 0.27 |
| 17 | 2.42 | 84.50 ± 0.33 |
| 18 | 2.06 | 84.65 ± 0.37 |
| 19 | 2.57 | 84.91 ± 0.34 |
| 20 | 2.32 | 84.89 ± 0.50 |
| 21 | 2.39 | 84.96 ± 0.25 |
| 22 | 2.68 | 85.03 ± 0.16 |
| 23 | 2.45 | 85.05 ± 0.25 |
| 24 | 2.65 | 85.02 ± 0.17 |
| 25 | 2.37 | 85.05 ± 0.13 |

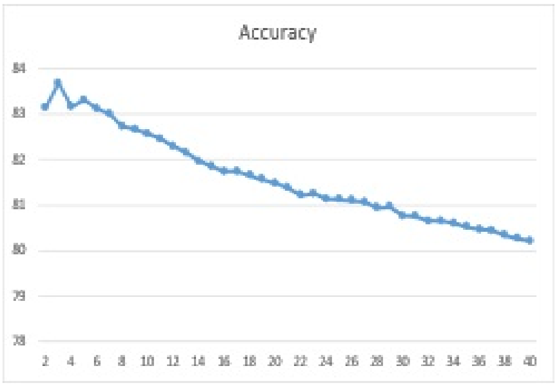


**Fig. 8** Graph of accuracy with respective to varying K values.

Highest accuracy is achieved at the variance smoothing of 1.00E- 6. The accuracy starts to plateau as the va_smoothing reaches 1.00E − 10. As the n_estimators are increased its accuracy tends to decrease. The highest recorded accuracy was achieved at n_estimators = 45 seen in Fig. 10.

To further optimize the model, hyperparameters were tuned using Optuna. A cutting-edge system for automated hyperparameter optimization. Optuna provides a define-by-run user API that enables dynamic search space construction as well as effective sampling and pruning algorithms. The standard tree structured Parzen estimator (TPE) Bayesian sampling procedure was employed. An objective function is defined, that takes the accuracy score of a tree trained and evaluated with the hyperparameters

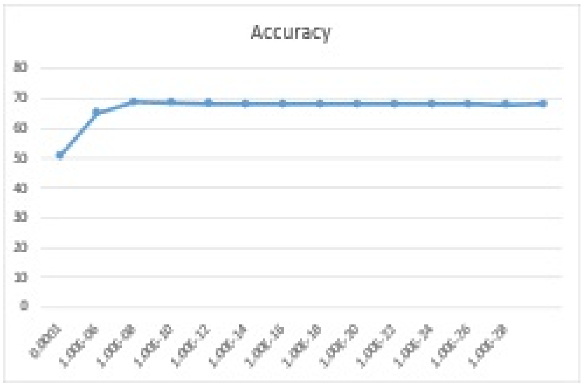


**Fig. 9** Accuracy graph of GaussianNB for varying curve smoothing values.

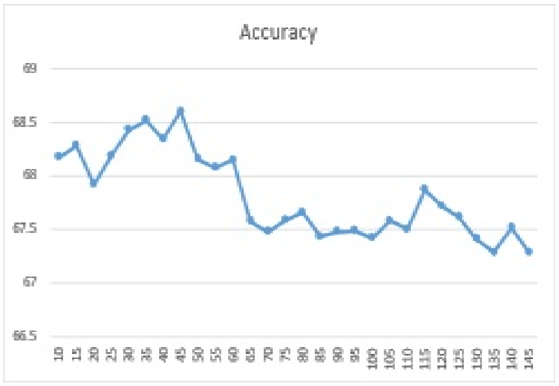


**Fig. 10** Accuracy graph for AdaBoost with varying n_estimators.

provided by the trial. Integer values for max_depth and min_samples_split are set to (20,27) and (2,5) respectively. Then, by using the optimize() method TPE was tuned, and 70 trials were executed. The best hyperparameters and score are shown in Table 3.

**Table 3** Multiclass classification accuracy achieved after implementation of Optuna

| Classifier | Accuracy (mean ± std) in% |
|---|---|
| Decision Tree | 85.26 ± 0.31 |
| KNN | 84.55 ± 0.11 |
| Random Forest | 88.31 ± 0.21 |
| Gaussian NB | 68.80 ± 0.80 |
| AdaBoost | 70.31 ± 0.57 |

## 3 CONCLUSION

Numerous categorization techniques from machine learning have been used, as detailed in this work, to determine the best method for identifying malware infections on Android devices. In this work, the potential of machine learning classifiers namely GaussianNB, AdaBoost, Logistic Regression, Random Forest, Decision Tree, and KNN is investigated to detect malware infections. The data source for the dataset was CiCMalmem-22. Random Forest and AdaBoost achieved a remarkably high accuracy of nearly 99.99%, respectively, of the samples properly identified in the trials. It is based on a 5-fold cross-validation for binary classification. This paper is also the only study that provides state of the art, multiclass classification of malware into their respective families. For multiclass classification of malware into Ransomware, Spyware and Trojan families, these classifiers were further tuned using Grid search and Optuna Hyperparameter Tuning techniques. In which, Random Forest hyperparameter tuned further by utilizing Optuna performed with highest accuracy, i.e., 88.31%. This also resulted in decreased computational times.

## DECLARATION

There are no conflicts of interests to declare.

## ACKNOWLEDGEMENTS

We would like to thank Vishwakarma Institute of Technology, Pune, for providing the ecosystem to work on research project.

## REFERENCES

[1] admin@enterpriseappstoday.com. "Enterpriseappstoday." (2010), [Online]. Available: `https://www.enterpriseappstoday.com/stats/android-statistics.html#:~:text=There%20are%203.3%20billion%20Android,The%20latest%20version%2C%20Android%2012.0`. (accessed: 15.05.2024).

[2] Z. Yuan, Y. Lu, and Y. Xue, "Droiddetector: Android malware characterization and detection using deep learning," *Tsinghua Science and Technology*, vol. 21, no. 1, pp. 114–123, 2016. DOI: `10.1109/TST.2016.7399288`.

[3] X. Liu, Y. Lin, H. Li, and J. Zhang, "A novel method for malware detection on ml-based visualization technique," *Computers & Security*, vol. 89, p. 101 682, 2020.

[4] M. Asam, S. J. Hussain, M. Mohatram, *et al.*, "Detection of exceptional malware variants using deep boosted feature spaces and machine learning," *Applied Sciences*, vol. 11, no. 21, p. 10 464, 2021.

[5] M. Brengel and C. Rossow, "Memscrimper: Time-and space-efficient storage of malware sandbox memory dumps," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, Springer, 2018, pp. 24–45.

[6] S. S. H. Shah, A. R. Ahmad, N. Jamil, and A. u. R. Khan, "Memory forensics-based malware detection using computer vision and machine learning," *Electronics*, vol. 11, no. 16, p. 2579, 2022.

[7] H. Safa, M. Nassar, and W. A. R. Al Orabi, “Benchmarking convolutional and recurrent neural networks for malware classification,” in *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, IEEE, 2019, pp. 561–566.

[8] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, “Novel feature extraction, selection and fusion for effective malware family classification,” in *Proceedings of the sixth ACM conference on data and application security and privacy*, 2016, pp. 183–194.

[9] T. Wüchner, M. Ochoa, and A. Pretschner, “Robust and effective malware detection through quantitative data flow graph metrics,” in *Detection of Intrusions and Malware, and Vulnerability Assessment: 12th International Conference, DIMVA 2015, Milan, Italy, July 9-10, 2015, Proceedings 12*, Springer, 2015, pp. 98–118.

[10] Ö. Aslan, M. Ozkan-Okay, and D. Gupta, “Intelligent behavior-based malware detection system on cloud computing environment,” *IEEE Access*, vol. 9, pp. 83 252–83 271, 2021.

[11] N. McLaughlin, J. Martinez del Rincon, B. Kang, *et al.*, “Deep android malware detection,” in *Proceedings of the seventh ACM on conference on data and application security and privacy*, 2017, pp. 301–308.

[12] R. Vinayakumar, K. Soman, P. Poornachandran, and S. Sachin Kumar, “Detecting android malware using long short-term memory (lstm),” *Journal of Intelligent & Fuzzy Systems*, vol. 34, no. 3, pp. 1277–1288, 2018.

[13] D. Zhu, Y. Ma, T. Xi, and Y. Zhang, “Fsnet: Android malware detection with only one feature,” in *2019 IEEE Symposium on Computers and Communications (ISCC)*, IEEE, 2019, pp. 1–6.

[14] H. Ma, J. Tian, K. Qiu, *et al.*, “Deep-learning–based app sensitive behavior surveillance for android powered cyber–physical systems,” *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5840–5850, 2020.

[15] M. S. Alam and S. T. Vuong, “Random forest classification for detecting android malware,” in *2013 IEEE international conference on green computing and communications and IEEE Internet of Things and IEEE cyber, physical and social computing*, IEEE, 2013, pp. 663–669.

[16] T. Carrier, “Detecting obfuscated malware using memory feature engineering,” 2021.

[17] T. Carrier., P. Victor., A. Tekeoglu., and A. H. Lashkari., “Detecting obfuscated malware using memory feature engineering,” in *Proceedings of the 8th International Conference on Information Systems Security and Privacy - ICISSP*, INSTICC, SciTePress, 2022, pp. 177–188, ISBN: 978-989-758-553-1. DOI: 10.5220/0010908200003120.

[18] K. M. Han J. Pei J., *Data Mining: Concepts and Techniques*. 2011.

[19] K. Alkhatib and S. Abualigah, “Predictive model for cutting customers migration from , banks: Based on machine learning classification algorithms,” in *2020 11th International Conference on Information and , Communication Systems (ICICS)*, IEEE, 2020, pp. 303–307.

[20] X. Pan, L. Zhu, Y.-X. Fan, and J. Yan, “Predicting protein–rna interaction amino acids using random forest based on submodularity subset selection,” *Computational biology and chemistry*, vol. 53, pp. 324–330, 2014.

[21] L. Rokach and O. Maimon, *Decision trees." Data mining and knowledge discovery handbook*. Springer New York, 2005.

[22] N. Ahmed, R. Ahammed, M. M. Islam, *et al.*, “Machine learning based diabetes prediction and development of smart web application,” *International Journal of Cognitive Computing in Engineering*, vol. 2, pp. 229–241, 2021.

[23] S. Shekhar, A. Bansode, and A. Salim, “A comparative study of hyper-parameter optimization tools,” in *2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, IEEE, 2021, pp. 1–6.

ITU Journal of Wireless Communications and Cybersecurity

ITU/JWCC

# Should Users Trust Their Android Devices? A Scoring System for Assessing Security and Privacy Risks of Pre-Installed Applications

**Abdullah Özbay**[1], **Kemal Bıçakcı**[2]

[1] TOBB University of Economics and Technology, Ankara, 06560, Turkey
[2] Informatics Institute and Computer Engineering Department, Istanbul Technical University, Istanbul, 34469, Turkey

**Abstract:** Android devices are equipped with many pre-installed applications which have the capability of tracking and monitoring users. Although applications coming pre-installed pose a great danger to user security and privacy, they have received little attention so far among researchers in the field. In this study, we collect a dataset comprising such applications and make it publicly available. Using this dataset, we analyze tracker Software Development Kits, manifest files and the use of cloud services and report our results. We also conduct a user survey to understand concerns and perceptions of users. Finally, we present a risk scoring system which assigns scores for smart phones consolidating our findings based on carefully weighted criteria. With this scoring system, users could give their own trust decisions based on the available concise information about the security and privacy impacts of applications pre-installed on their Android devices.

**Keywords:** Mobile security, privacy, Android, pre-installed apps, scoring system.

## Kullanıcılar Android Cihazlara Güvenmeli mi? Ön-yüklü Uygulamaların Güvenlik ve Gizlilik Risklerini Değerlendirmek İçin Bir Puanlama Sistemi

**Özet:** Android cihazlarda, kullanıcıları izleme ve gözlemleme yeteneğine sahip birçok ön-yüklü uygulama bulunmaktadır. Ön-yüklü uygulamalar kullanıcı güvenliği ve gizliliği için büyük bir tehlike oluşturmasına rağmen, şimdiye kadar bu uygulamalar araştırmacıların kısıtlı ilgisini çekmiştir. Bu çalışmada, böyle uygulamaları içeren bir veri kümesi oluşturduk ve bunu herkese açık hale getirdik. Bu veri kümesini kullanarak, takipçi Yazılım Geliştirme Kitleri, manifest dosyalarını ve bulut hizmetlerinin kullanımını analiz ettik ve sonuçlarımızı raporladık. Ayrıca, kullanıcıların endişelerini ve algılarını anlamak için bir kullanıcı anketi gerçekleştirdik. Son olarak, bulgularımıza dayanan dikkatlice ağırlıklandırılmış kriterlere dayalı olarak akıllı telefonlar için risk puanlama sistemi sunuyoruz. Bu puanlama sistemi ile, kullanıcılar Android cihazlarındaki ön-yüklü uygulamaların güvenlik ve gizlilik etkileri hakkında mevcut bilgilerini kullanarak bu uygulamalara güvenip güvenemeyeceklerine karar verebilirler.

**Anahtar Kelimeler:** Mobil güvenlik, mahremiyet, Android, ön-yüklü uygulamalar, skorlama sistemi.

RESEARCH PAPER

Corresponding Author: Abdullah Özbay, a.ozbay@etu.edu.tr

Reference: A. Özbay, K. Bıçakcı, (2024), “Should Users Trust Their Android Devices? A Scoring System for Assessing Security and Privacy Risks of Pre-Installed Applications,” *ITU Journ. Wireless Comm. Cyber.*, 1, (1) 9–28.

Submission Date: Apr, 04, 2024
Acceptance Date: Sept, 09, 2024
Online Publishing: Sept, 30, 2024

## 1 INTRODUCTION

Android is the most widely used mobile operating system [1] in the world mainly due to two reasons: (i) it is an open-source operating system [2], (ii) Google makes manufacturers' job of producing new devices much easier if they prefer Android [3]. Not only manufacturers, but also mobile network operators, semiconductor producers and third party companies that assist and collaborate with manufacturers can easily modify and add their own applications to mobile devices with Android.

Google provides certification programs auditing Android devices, firmware and pre-installed applications. In the Android Compatibility Program, Android Compatibility Definition Document [4] is used to check for device and firmware compatibility. The requirements can be checked using Compatibility Test Suite [5]. However, in this program, there is no privacy and security audit applied to an Android device.

Google also offers Android Certified Partners Program [6] to device manufacturers. Device manufacturers have to satisfy this program's requirements to be a Android Certified Partner [7]. As part of this program, mobile Built Test Suite (BTS) [8], Security Test Suite (STS) [8] and some other suites are applied. Within BTS, Potential Harmful Applications (PHAs) and other harmful actions are examined. Also, in STS, security patches are checked to verify that pre-installed applications are up-to-date. But, neither Android Compatibility Program nor Android Certified Partners Program guarantees security and privacy of users.

In real life, many pre-installed applications threatening security and privacy of users have been already detected. One of the well-known examples is Adups discovered by Kryptowire [9]. Adups is a Firmware Over The Air (FOTA) application that helps manufacturers to update device firmware remotely. According to the analysis, this application that exists in BLU R1 HD smartphones has the ability to collect Personally Identifiable Information (PII) and run privileged code on user's devices.

As stated in Google's Android Security & Privacy 2018 Year In Review [10], Android smartphones could be infected with ease since developers of a PHA need to deceive only one of the OEMs (Original Equipment Manufacturers) or other companies in the supply chain for the installation. There were several PHAs detected in smartphones in big Android markets such as India, USA, Brazil and Indonesia. Furthermore, researchers from Oversecured have found that pre-installed applications on Samsung devices certified in Android Certified Partner Program have multiple dangerous vulnerabilities [11]. We also note that third party applications that are not directly related with OEMs e.g., social networking, search engine, news, telecommunication, etc. may also be pre-installed in Android smartphones. For example, as reported by Bloomberg [12], Facebook apps are pre-installed and cannot be deleted from smartphones. These third party applications and their affiliated companies usually cooperate with manufacturers [13].

Until recently, studies on pre-installed application ecosystem analyze only a couple of selected applications and pre-installed applications in mobile devices did not attract much attention from researchers. However, with a recent study [14] on pre-installed Android software, the gap has begun to close. On the other hand, there are many aspects of pre-installed applications that has not been explored yet. In this paper, we identify and complete the missing spots on previous work, as described next.

First of all, because there is no public data set which consists of pre-installed Android applications (We contacted the authors of previous work [14], but they informed us that sharing their dataset is not possible), our first aim is to make such a dataset available. We believe this dataset could facilitate further research on this important topic. For this purpose, we implement an Android application (Pre-App Collector) and use it to collect pre-installed applications from the devices of volunteers. As stated in Pre-App Collector's user consent screen [15], we do not access, collect, share or analyze any kind of personal data. The data being made publicly available does not disclose any personal data.

Regarding user privacy, using the collected data set, we extract tracker SDKs from applications. Then, we analyze the goals of these trackers which could be analytics, advertisement, location tracking, profiling, identification, etc. Also, we check what kind of applications (OEM, mobile network operator, social networking, etc.) contain these trackers. This analysis is the first attempt to discover tracker SDKs ecosystem on pre-installed Android applications and the effects of trackers on user privacy.

From security point of view, we make the first study in literature on critical fields of manifest files in pre-installed applications. Within this scope, we investigate exported application components, shared UIDs, attributes such as *usesCleartextTraffic, allowBackup* and *debuggable* in manifest files and find out that if pre-installed applications follow Android security best practices. In addition, we search cloud services used by Android pre-installed applications. By doing so, we intend to find out that how securely these apps take advantage of these services.

In addition, we make a survey (with users who download and use our application [15]) to understand their concerns and perceptions regarding security and privacy of pre-installed applications. Finally, we make a comprehensive evaluation of pre-installed applications from security and privacy point of views using multiple criteria based on both our and earlier findings and present a device scoring system. Device scores aim at making our findings more understandable for average users of smart phones.

To summarize, with this study we contribute to the young literature of pre-installed mobile applications and their security and privacy implications in following ways:

- We discover tracker SDKs ecosystem that exists in Android pre-installed applications.
- We analyze manifest files of applications to check compliance to security best practices.
- We analyze cloud services that are used by pre-installed applications and check if any misconfiguration exists in these services.
- We report the results of a survey applied to users who install our application [15] to shed light on user concerns and perceptions regarding security and privacy of pre-installed applications.
- We make our preinstalled app dataset publicly available [16]. The detailed metadata information about these files is available in our website [17].
- We present a scoring system to make the results of our analysis more understandable by average users. We publish our analysis results and device scores on a website [17] to inform users and researchers.

The rest of the paper is organized as follows, Section 2 summarizes the results of earlier studies on the topic. Section 3 presents our Android application developed for collecting data on pre-installed apps and provides general information about the dataset made available. Section 4 describes the analyses we perform and presents the results we obtain. Section 5 contains user survey results and related discussion. Section 6 includes the details of our scoring system and the remarks on the scores of some devices. Section 7 lists the limitations of this study. Finally, Section 8 concludes this paper.

## 2 RELATED WORK

There are many previous studies on applications available at Android Application Markets (e.g., [18], [19], [20], [21], [22]) as opposed to being pre-installed. A considerable portion of these focused on application permissions due to their importance with respect to user privacy and security [23], [24], [25]. Custom permissions were also studied [26]. We note that when applications are pre-installed, users do not have the chance to grant or deny dangerous application permissions [27] as they can normally do.

Third-Party Libraries (TPLs) like SDKs are crucial for Android application development as they help developers to expedite application development process. However, these TPLs may contain codes that are related to advertising and tracking services. Earlier studies [28], [29], [30], [31] found out that these services threaten user privacy.

Misconfigurations in Android application manifest files and cloud services used by applications can cause privacy and security issues. Two recent studies [32], [33] which focused on cloud service misconfigurations indicate that unsecured cloud services may expose personal data. In addition, manifest file attributes (e.g., *allowBackup, debuggable, usesCleartextTraffic*) and shared UIDs should be configured carefully as specified in the guidelines [34]. Particularly, intentional or unintentional misuse of shared UIDs may lead to over-privileged (e.g., with *android.uid.system privilege*) execution of applications [8]. Additionally, applications that have the same shared UIDs and signed with the same keys may access each other's resources. This can lead to situations which affect security and privacy of users [35], [36]. Even though there are significant advances on standardization of secure application development [37], as we observe in our work, they are not widely adopted yet in practice.

As already mentioned, most earlier work cover Android applications from Android Application Markets. Since pre-installed applications come with devices, require no further installation and most of them have more privileges beyond those available to standard developers, they demand a more elaborate and focused analysis. The effects of so called bloatware applications that come pre-installed and waste system resources like battery, disk space, memory etc. were investigated in a recent paper [38]. This paper also includes a user study conducted to understand users' knowledge and awareness regarding bloatware applications. But, it mostly focused on application permissions and their consequences. There is also a study [39] that aims to find privilege escalation vulnerabilities of pre-installed applications using taint analysis methods. In another recent study [40], pre-installed OTA applications were studied.

In another recent study [14], an analysis of pre-installed applications was presented. Although their analysis is the first large scale study on the subject, the authors admit that they were only able to scratch the surface of a much larger problem. We see that their analysis was mostly limited to third party libraries, application permissions (particularly custom permissions) and network traffic of applications.

As stated so far (and summarized in Table 1), pre-installed applications and applications from app markets differ substantially. We definitely need a better understanding of the pre-installed app ecosystem and its security and privacy implications. Our goal in this paper is to contribute in this regard and the list of our contributions is provided at the end of section 1 [1].

[1] Preapp Collector app [15], which we developed independently, has a user interface and functionality comparable to the application used in [14]. But there is no repeat of analysis on the collected data set in our work, which focuses on previously unexplored aspects of pre-installed applications. On the other hand, to obtain a more comprehensive scoring system, we also consider the results of earlier work [14] as further discussed in section 6.

**Table 1** Comparison of pre-installed and app market applications

| **Pre-installed Applications** | **App Market Applications** |
|---|---|
| Pre-installed on devices | Installed by the user from App Markets |
| Runs with more privileges | User privileges |
| Mostly cannot be uninstalled, only disabled | Can be uninstalled |
| Updated less frequently | Updated more frequently |
| Permissions mostly automatically granted without user consent | User consent required for permissions |

This paper is an extension of work originally presented in Turkish in a conference [41]. The conference paper contains essentially only a condensed and early version of our tracker analysis and user study. This paper not only presents a more elaborate discussion on these parts, but also extends our work with new sections i.e., security analysis (subsection 4.2), scoring system (section 6) and limitations (section 7).

## 3 PRE-APP COLLECTOR AND DATASET

In this section, we provide information about the application we develop to collect the dataset and share general statistics and some early analysis results regarding this dataset.

### 3.1 Android Application (Pre-App Collector)

Up to our best knowledge, no public dataset that consists of Android pre-installed applications exists. A recent study [14] has created such a dataset, but it is not publicly available. Therefore, we decide to prepare our own dataset and make it publicly available [16]. For this purpose, we implement an Android application to collect the pre-installed app data from user's devices. Our study was approved by TOBB University of Economics & Technology Human Resource Evaluation Board [42]. We make this application available on Google Play Store [15]. To announce the application, we use e-mail groups from universities, social media groups, and also share it on social media.

The application works as follows. When it starts, we inform users about our study, take their consents to start the data collection and ask a couple of questions as part of our survey to understand their concerns and perceptions regarding security and privacy of pre-installed applications. The data collected about the device includes data of manufacturer, model, product, version, timezone, SIM operator, SIM country. Then, we scan /system, /odm, /oem, /vendor, /product directories recursively to reach firmware files including pre-installed applications. Hash of these files are calculated and sent to our server to check if they already exist in our dataset. The list of files that are not in our dataset is sent to the device so that these files are also transferred to our server. Finally, we show users a summary containing the list of pre-installed applications and statistics about firmware files.

### 3.2 General Statistics

We present the basic statistics about our dataset as follows:

- We collect files from 22 different OEMs and 98 different devices (We distinguish non-identical devices using unique ID values. On the other hand, these values cannot be used to uniquely identify users and their devices).
- We determine using timezone information that users from at least 14 countries have installed our application.
- In total, we collect 143862 firmware files including 14178 apk files, 418 certificates and 58721 libraries.
- In total, 77 users participate in the survey (excluding survey results that have the answers as default picks or do not have a proper e-mail address).

### 3.3 Early Analysis and Its Results

We perform a number of early analysis. First, we use Androguard [43] which is a Python based Android reversing tool to extract certificates that are used to sign the applications [44]. We analyze the so-called Issuer field in application certificates to detect which person or company developed the application. We group these certificates because not always a single certificate is used to sign the applications developed by the same entity. We specify groups considering OEMs, OEM-related, and Third Party information (e.g., Social Networking, Web Browser, Application Marketing, Caller Identification, News, Dictionary, Cloud Service, Telecommunication Companies, Marketing & Advertising Services, etc). In total, we determine 126 certificate groups and applications under these groups.

In addition, we check what portion of determined pre-installed applications exists in Google Play Store [18] using application package name. We find out that only 9% of the applications can be accessible from Google Play. Moreover, while collecting the applications, we also obtain metadata about apps e.g., first install time and last update. The analysis of this metadata shows that 7829 out of 14178 (55%) pre-installed apps were not updated ever since they came with the devices. We note that because most of the pre-installed applications are not third party ones and located in the system partition, they can only be updated by over-the-air update mechanism released by vendors and require smartphones to be restarted. Thus, a pre-installed application cannot be easily updated like the applications from app markets.

## 4 ANALYSIS

We analyze pre-installed applications with respect to impacts on both user privacy and security, as detailed below.

### 4.1 Privacy Analysis

In privacy part, we perform a detailed analysis of tracker SDKs and privacy policies.

**Tracker SDKs**. Android tracker SDKs collect data about users and how they use applications. They may be embedded to pre-installed applications and have various functionalities like crash reporting, analytics, profiling, identification, advertisement, location tracking. To analyze trackers, we base our study on the work by Exodus Privacy [45], a non-profit organization working on Android trackers and their effects on user privacy. We take advantage of their tool named *exodus-standalone* [46] to detect embedded trackers in pre-installed applications. As a result, we discover tracker ecosystem and their effects to user privacy in pre-installed applications. Our early findings could be summarized as follows:

- 85 different trackers installed in 836 different applications were detected.
- We examined privacy policies of companies which use the trackers and noticed that some of them do not clearly state what kind of information they collect. (When they do not provide multi-language support, we use online translation services to investigate them.)
- In their privacy policies, most trackers stated that they track sensitive information such as PII, location-related data, log information, user behaviour, device identifiers and advertisement IDs (e.g., Google Advertising ID [47]). This practice threatens user privacy at different levels.
- Most of the trackers stated that they comply with regulations like GDPR [48] and CCPA [49], but still a few do not mention them in their privacy policies. Trackers tend to collect more data when they are not under these regulations.

**Tracker Statistics**. As stated above, we detected so many trackers in so many different apps. Some of these trackers are more common than the others in pre-installed applications. In Figure 1, we list the most common trackers that exist in pre-installed applications. It is not surprising to see that big technology companies such as Google, Facebook, Tencent and Amazon are dominant here.

Also, we observe that a number of applications come with excessive number of trackers which arguably makes violation of user privacy inevitable. Figure 2 lists application package names which have the highest number of tracker

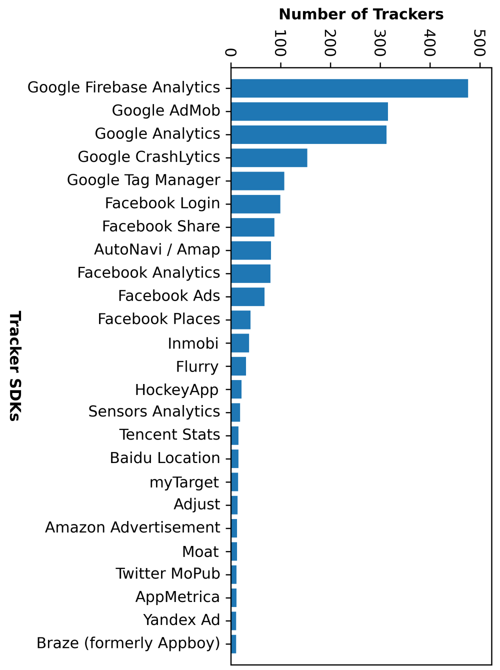


**Fig. 1** Most common tracker SDKs in pre-installed applications.

**Table 2** Companies and the number of tracking services related with them

| Company | Number of Tracking Services |
|---|---|
| Alphabet (Parent Company of Google) | 5 |
| Facebook | 5 |
| Oath | 3 |
| Baidu | 3 |
| Microsoft | 3 |

SDKs (different versions are considered as the same application). Interestingly, most of these applications are third party applications according to our certificate based analysis. Consequently, the devices do not actually require them to work properly.

In addition, we group trackers based on their companies. Some of the tracking services are offered by companies affiliated with big technology companies. In Table 2, we list these companies and the number of tracker-related companies that are affiliated with them.

According to our analysis (see Table 3), big technology companies acquire tracking services continuously. Once we check companies offering tracking services from Crunchbase [50], a website that provides data about companies and the people behind them, we noticed that tracking companies are acquired by other technology companies aiming to grow and expand their market share. This situation brings additional privacy risks because some tracking services state in their privacy policies that once they are ac-

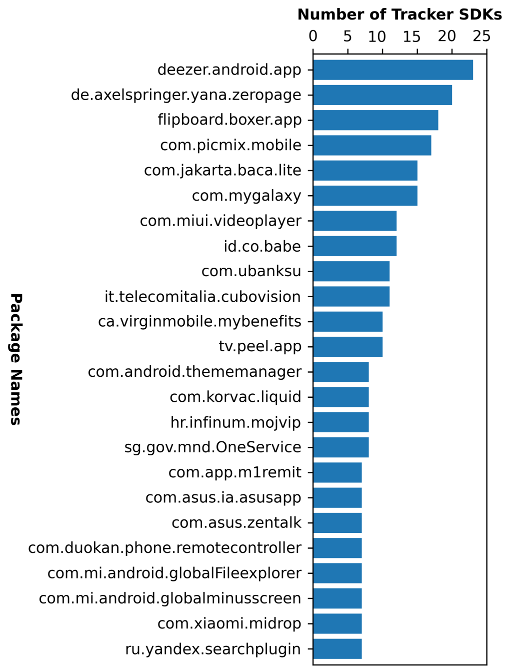


**Fig. 2** Applications that contain the highest number of tracker SDKs.

**Table 3** Number of tracker companies in different countries

| Country | Number of Companies |
|---|---|
| United States | 56 |
| China | 11 |
| Russia | 4 |
| Germany | 4 |
| France | 3 |
| India | 2 |
| United Kingdom | 1 |
| Israel | 1 |
| Open Source | 1 |

quired by another company, user data becomes no longer under their control and is shared with this company.

Finally, we checked the headquarters of these companies from Crunchbase. Table 3 shows the number of tracking companies located in different countries. We note that some of these countries such as Russia and China are not under any well-known regulations (e.g., GDPR, CCPA) protecting user privacy.

**Purpose of Trackers**. Tracker SDKs may provide different functionalities as they are designed for different purposes. Hence their impact on user privacy varies accordingly. Exodus Privacy [45] categorizes tracker SDKs in six groups:

- **Crash Reporters**: The goal of these trackers is to notify developers when applications crash.
- **Analytics**: This kind of trackers collect usage data and enable developers to learn about the users. For example, browsing behaviours are collected.
- **Profiling**: By collecting from users as much data as possible, these trackers try to build virtual profile of users. For this purpose, trackers collect data like browser history, list of installed applications, etc.
- **Identification**: The purpose of these trackers is to specify users' digital identity. Developers may associate online activities of users with their offline activities.
- **Advertisement**: The aim of these trackers is to show users targeted advertisements by using users' digital profiles and help developers to monetize their applications.
- **Location**: These trackers are used to locate users by taking advantage of Bluetooth, GPS antenna, IP address, etc.

We categorize trackers we have detected using this grouping since the effects on user privacy varies per group. Figure 3 shows the number of trackers associated with each group. As stated, each tracker group has a different functionality (some trackers perform more than one functionality). On the overall, trackers under analytics, profiling and identification groups highly threaten user privacy since they mostly need to collect personal data to fulfill their functionality. Location trackers collect location data which is also sensitive. Advertisement trackers access and collect personal data for a targeted advertisement, which might also have privacy implications. However, not all trackers are evil, crash reporters mostly do not threaten user privacy. As mentioned, they are mostly used to report application failures to help developers.

**Privacy Policies.** We investigate tracker companies' privacy policies and related privacy issues to understand what kind of data is collected, what data is shared with whom and whether or not these companies comply with regulations such as GDPR and CCPA.

Privacy policies confirm that all of the trackers without exception collect various types of user data. Below, we present interesting points in privacy policies of tracker companies regarding their data collection routines.

First of all, most of the tracking services collect location data in various ways. For instance, nearly all services collect IP addresses, using this information approximate location of users can be determined. Also, when available, services might access GPS data from the device to locate users. Moreover, a few of the trackers collect nearby Wi-Fi hotspots, cellular and Bluetooth information to produce the most precise location information.

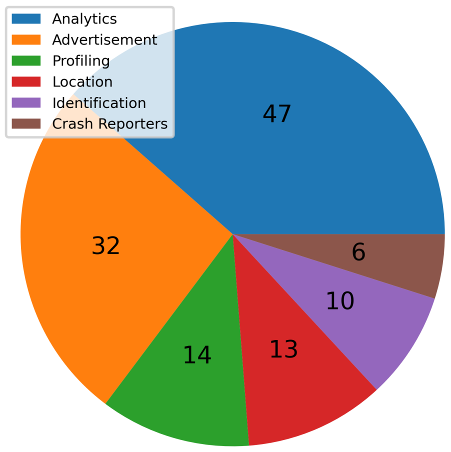


**Fig. 3** Total number of trackers detected per each tracker group.

Secondly, nearly all of the trackers access advertisement IDs such as Google Advertising ID to recognize devices for advertisement purposes.

Thirdly, many of the trackers collect information about network connections such as MAC addresses, connection types (e.g., Wi-Fi, cellular) in addition to IP addresses. This information is beneficial both for location tracking and device identification.

In addition, some tracking services collect device identifiers like IMEI and IMSI numbers. This kind of data cannot be changed by users and can be used to identify the devices. The risk due to IMEI number collection is well-known [51].

Furthermore, to profile users, a number of tracking services collect information such as browser history, application log, application usage stats, cookies, etc.

Finally, some of these companies collect Personally Identifiable Information (PII) such as name, email address, gender, contact information (e.g., telephone number), etc.

Below, in order to embody the associated privacy risks, we compile a small subset of real life cases concerning trackers:

- Behavioral analytics company named Sensor Analytics, whose owner is Sang Wenfeng, former technology manager at Baidu Inc's big data department, has a partnership with Xiaomi [52] to work on tracking users.
- Citizen Lab claims that Baidu Mobile Analytics SDK causes sensitive data leaks [53]. The leak data include IMEI number, GPS location and nearby wireless access points. In addition, Baidu Map service may collect sensitive data such as IMEI number, IMSI number, MAC address, etc. [54].
- According to a research by Gizmodo, applications that use Bugly crash reporting service collect and send IMEI numbers and IP addresses to servers located in China [55].
- As stated in its privacy policy, Chinese tracking service Mintegral may collect IMEI numbers of users. Also, it cooperates with advertisement exchange platform like Google DoubleClick, Inmobi, MoPub, Tencent, Baidu, etc. [56].
- From the applications that embed its tracking code, MoEngage may obtain PII like email address, name and phone number as indicated in its privacy policy [57].
- Applications that use JPush service may send IMEI numbers, MAC addresses, serial numbers, and precise location data to Aurora Mobile's servers [58].

**Data Sharing**. Analysis of privacy policies shows that tracking services may share data collected from devices. In general, the data may be shared with:

- Affiliates and Subsidiaries,
- Service Providers,
- Law Enforcement Units,
- Business transfers,
- Advertisers,
- Researchers and Academics,
- Publishers,
- Data Partners.

Also, as pointed out in a study on tracking ecosystem [28], all of the ten largest tracking organizations could share collected data with third parties and subsidiaries. Because of these sharing routines, opt out chance of users is in danger since different companies have different opt out procedures. Moreover, tracking companies may share data with each other e.g., MoPub's partnership with Integral Ad Science, DoubleVerify and Moat [59].

Lastly, to the best of our knowledge, all of the tracking companies share data for legal purposes (e.g., law enforcement requirements). Even if this stems from a good intention to help law enforcement units, it can be abused by some governments [60].

**Compliance with Regulations**. Under the protection of regulations like CCPA, GDPR and COPPA, users have more control over their data. They can learn what kind of data is collected, with whom their data is shared or to whom it is sold, etc. Our analysis on privacy policies show that when companies are not required to comply with these regulations, they are more likely to ignore privacy rights of users (e.g., without these regulations, as we saw in Mintegral example [56], companies continue to abuse their capabilities). High fines probably obligate the companies to adapt to these regulations and show more respect to data privacy.

## 4.2 Security Analysis

We analyze security practices in manifest files (*AndroidManifest.xml*) and cloud service configurations of applications.

**Manifest File Analysis.** A manifest file is an XML file that describes application specific essentials [61] containing app's package name, app components (activity, service, broadcast receiver, content provider), app permissions, app attributes and manifest attributes. We examine attributes such as *sharedUserId, allowBackup, usesCleartextTraffic, debuggable*, which are among the most critical fields with respect to user security. Below, we explain the security implications of misconfigurations in these fields together with our findings on the dataset.

**sharedUserId**. In Android, unique user ID values are assigned to each application. However, in some conditions, for instance, when the same developer or company have multiple applications on a smartphone and want to share application resources (e.g., permissions, code) with each other, the same user ID value may be assigned to these applications. For this functionality, *sharedUserId* attribute is used. But misconfiguration of this attribute may cause security vulnerabilities. Also, adversaries could take advantage of this attribute to hide their malicious codes from security analysts (because of the risk this attribute brings, it was deprecated in API level 29 by Android).

Pre-installed applications that are signed as system apps with the same certificate can run with system user privileges, one of the most privileged users in Android system. We observed that 3303 out of 14178 pre-installed applications possess shared UID value of *android.uid.system* which gives system privileges to applications. Vulnerabilities in these applications may cause adversaries to access devices with the system privileges [62]. Also, malware (e.g., Adups malware [9]) may be embedded with system privileges in devices as we have mentioned. In our analysis, we detected apps that run with system privileges without a real need (e.g., *com.caf.fmradio*). Clearly, this practice violates the least privilege principle.

**allowBackup**. When this attribute is valid in the manifest file and if USB debugging is enabled in an Android device, application data can be backed up by anyone who has physical access to it. Thus, all data in */data/data/package_-name* can be exported from the smartphone. If any unencrypted sensitive data such as PII, passwords, keys etc. is stored in such a directory, adversaries who has physical access may easily capture it.

We examined if any application has enabled *allowBackup* attribute. We also analyzed its prevalance in each certificate group. We detected 6847 applications in total that allow backup using adb [63]. In Figure 4, vendors with the highest number applications in this configuration are presented. Almost all vendors have enabled the *allowBackup* attribute. We think this practice requires further investigation due to its security implications.

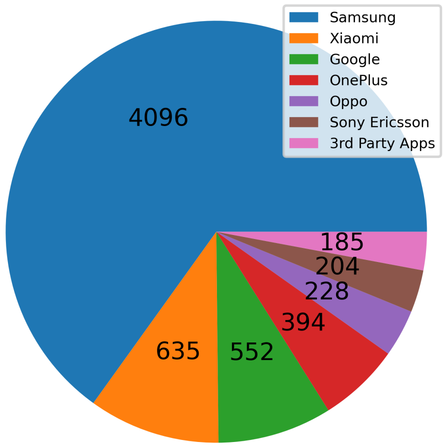


**Fig. 4** Vendors and the number of pre-installed applications signed by them in our dataset that allow data backup.

**usesClearTextTraffic**. Applications may use cleartext traffic to connect to remote servers. This can cause private and sensitive data to be eavesdropped by adversaries [64]. With Android 6.0, application developers may prevent their applications to send cleartext data by configuring the *usesClearTextTraffic* attribute. However, we detected a considerable number of pre-installed applications with the *usesClearTextTraffic* flag set to "true". 1270 of the apps from our data set may send their data as cleartext to servers. Most of these apps are belong to OEMs and only 37 of them are belong to third parties.

**debuggable**. We also analyzed the configuration of *debuggagle* in pre-installed applications. When this flag is enabled in an application, it may be debugged by users who have physical access to the device with tools like jdb [65]. Using this functionality, classes and functions of apps can be easily read and even manipulated. In addition, it is possible to execute arbitrary code within the permission context of these applications. Thus, it is strongly suggested that to set this flag as "false" in production code. Fortunately, we only found 5 such applications (three variants of *com.sec.android.kiosk*, *com.trendmicro.mars.mda.httpserver*, and *com.huawei.camera2.mode.cosplay*). It was surprising to see that the later application was signed by the Android Debug Certificate [44]. Most app stores does not accept applications that are signed with this type of certificate (recall the difference between pre-installed and app market applications).

As the conclusion of manifest file analysis, we state that the best practices regarding the use of attributes are not embraced satisfactorily by the developers of pre-installed applications.

**Use of Cloud Services** Almost all Android applications connect to backend servers to fulfill its functionality. These servers can be used for various purposes such as storing data, querying for information, performing actions for application, etc. Not every developer or company has the resources and time to implement their own server infrastructure. Even when they have enough resources, they might choose not to use their own server because cloud based solutions are easy to manage and provide many other advantages. These solutions offer functionalities such as data storage, notification management, analytics, API based services, etc. Due to their critical role in the mobile app ecosystem, cloud-based solutions need to be managed carefully in terms of security and privacy. Although they may be regarded as secure by default, developers should still be aware of their correct configurations and operation logic before using them. Unfortunately, considerable number of developers overlook the configuration of these solutions, that may affect millions of people.

Some of the popular cloud-based services are Google Firebase [66], Amazon Web Services (AWS) [67], Microsoft Azure [68] and Google Maps API [69]. Pre-installed applications also heavily use these services. We examined use of cloud services by these applications and misconfigurations exist in them.

These services require special keys, secrets and URL formats. Disclosure of these values may cause unauthorized access to company resources, sensitive and confidential information leaks, denial of service attacks and waste of company resources. To see whether we could extract these values, we took advantage of several tools [70, 71, 72, 73] and also wrote a few custom scripts. We also manually analyzed some of the applications by reverse engineering. As a result, we detected vulnerabilities related to Google Maps API, AWS, Firebase, Slack Webhooks [74], and OAuth [75]. Using custom Python scripts, we tested and validated our findings. Below, we discuss interesting results with respect to user security and privacy.

**Google Maps API**. Using this API service, developers could retrieve location-based data. Until 2018, this service was free. However, in June 2018, Google launched the pay-as-you-go pricing model [76]. In this model, the price is determined according to the number of request that is made to Product Stock-Keeping Unit (SKU) [77]. A SKU is a combination of Product API and the service or function called e.g., Place API - Photos Details.

To test whether Google Maps API key values are extractable, we used a modified version of *apkleaks* [70], a Python tool that uses special regex patterns for various URIs, endpoints and secrets for mass file scan. We tested the extracted keys using a modified version of *gmapsapiscanner* [73] so that unauthorized accesses using these keys can be verified. We present our results in Table 4 that consists of *Name of Vulnerable SKU*, *Vulnerable Application Count* that use SKU and *Impact(s)* of vulnerability that exists in SKU. These vulnerabilities may cause waste of monthly quota. Adversaries may also conduct denial of service attacks if there is a maximum bill limit.

**Amazon Web Services**. Since Amazon Web Services (AWS) cloud computing platform is widely used by mobile application developers and companies [78], we expect that it draws attention of attackers more than others. Mobile applications utilize Amazon Simple Storage Service (S3), which is subsidiary service of AWS to store various objects. In Amazon S3, the key concepts are Buckets, Objects, Keys and Regions. Bucket is a kind of container used to store and organize objects. Object is a fundamental entity consists of object data and metadata. To identify each object, Key is used. Finally, Region shows in which geographical region buckets are stored. For example, in the URL *https://awsexamplebucket1.s3.us-west-2.amazonaws.com/photos/puppy.jpg*, *awsexamplebucket1* is the name of the bucket, *photos/puppy.jpg* is the object and *us-west-2* is the region.

In Android ecosystem, developers need API keys (AWS access key ID and AWS secret access key) to access buckets and store objects in these buckets. Disclosure may allow adversaries to access Amazon S3 buckets and objects. Amazon has a documentation [79] that contains the best practices for managing these keys. Accordingly, these keys should not be embedded in application code directly, instead they should be stored at places suggested by Amazon or developers should use the Token Vending Machine [80]. Also, they should be renewed periodically for security reasons.

In our analysis, we detected plenty of S3 buckets, AWS access key IDs and AWS secret access keys by using *apkleaks* tool and/or by manual reverse engineering of apk files. We found a number of key pairs useful to access Amazon S3 buckets automatically. We tested them to see if any of them are still valid and can be used to access S3 buckets. We verified that accessing buckets of at least two different companies was possible. The number of valid key pairs we have found is not many but the impact could be outrageous. Using these keys, it was easy to access S3 buckets of companies which reveal not only the application information but also buckets and objects of various other applications and services. This situation clearly violates the principle of least privilege. In addition, we investigated the objects in these buckets and confirmed that sensitive information such as PII, credentials and source code of applications and ser-

**Table 4** The number of vulnerable applications in our dataset for different Google Maps API SKUs

| Vulnerable SKU | Vulnerable Application Count | Impact(s) |
|---|---|---|
| Places Photo API | 199 | $7 per 1000 requests |
| Nearby Search-Places API | 198 | $32 per 1000 requests |
| Text Search-Places API | 198 | $32 per 1000 requests |
| Find Place From Text API | 196 | $17 per 1000 elements |
| Autocomplete API | 196 | $2.83 per 1000 requests, Per Session - $17 per 1000 requests |
| Place Details API | 196 | $17 per 1000 requests |
| Staticmap API | 161 | $2 per 1000 requests |
| Geocode API | 81 | $5 per 1000 requests |
| Geolocation API | 51 | $5 per 1000 requests |
| Timezone API | 36 | $5 per 1000 requests |
| Embed (Basic) API | 26 | Free |
| Elevation API | 16 | $5 per 1000 requests |
| Streetview API | 15 | $7 per 1000 requests |
| Embed (Advanced) API | 12 | Free |
| Directions API | 7 | $5 per 1000 requests, (Advanced) - $10 per 1000 requests |
| Distance Matrix API | 5 | $5 per 1000 elements, (Advanced) - $10 per 1000 elements |
| Nearest Roads API | 4 | $10 per 1000 requests |
| Route to Traveled API | 4 | $10 per 1000 requests |

vices could be accessed. We contacted to the companies that developed these applications about the discovered vulnerabilities via e-mail. One of them responded by confirming this vulnerability and stated that the concerned application is no longer supported by them. The other company did not respond to our e-mail. As we notified the vendors about these vulnerabilities more than a year ago, we report them in this paper in a responsible manner (without identifying them). We urge developers to use these keys securely and be aware of impacts of their disclosure.

**Google Firebase Database**. Google offers developers and companies a cloud based database [66] to store their data in JSON format. This database, named as Firebase Realtime Database, can be used via SDK and has some key capabilities i.e., real-time synchronization, offline response management, multiple database scalability, direct access from different clients (mobile device, web browser). To utilize this database, developers should create a database from the Firebase console. This database is named as *<database-name>.firebase.io* or if region is supplied as *<databaseName>.<region>.firebasedatabase.app*. By default, anyone can access it, hence Firebase database should be configured properly to prevent unauthorized read and write accesses.

In our work, using the *apkleaks*, we detected Firebase URLs in applications with the pattern mentioned above. We found 665 applications using Firebase databases and tested them using a custom Python script. To see if a database is readable by anyone, we simply add ".json" at the end of database URL and check the status code of the response which is 200 when readable. In addition, to find the world-writable databases, we send a put request to the database URL together with some JSON data and check the status code of response whether it is 200 or not. As a result, we found two Firebase databases that belong to two different applications everyone may read and write. Fortunately, there was no sensitive or confidential data which belong to users or companies. Developers and vendors should be careful about the Firebase database configuration as they may contain sensitive data of users and companies in other use cases.

**OAuth**. With *client_id* and *client_secret* values, Android applications generally use OAuth 2.0 to access different APIs or services. These values (especially *client_secret* value) should be protected against unauthorized access. For better protection, developers should take advantage of *Proof Key for Code Exchange (PKCE)* flow in which the client creates a new secret on each authorization request and uses this secret when exchanging authorization code for an access token [81]. However, in our analysis, we observed many applications that store static OAuth values belonging to services such as Google, AOL, Outlook, Office 365, Yahoo, Microsoft and mail.ru as cleartext. Thus, attackers can steal these values and use them to access APIs or services.

## 5 USER SURVEY

While getting help from users installing our app for building our dataset [15], we also asked them several questions to shed light on their concerns and perceptions about pre-installed applications as well as their general attitude toward smart phone usage and choices (Survey questions are provided in the Appendix APPENDIX A).

77 users attended to our survey (we eliminate results with answers all as same as the default ones and the results with an email address that has been previously used). At the be-

ginning, we asked questions on demographics. There were 40 participants in 25-34 age range and 19 in 18-24 age. 25 were female with one person chose not to provide gender information. Educational level of participants is at least Bachelor degree (70%). Only 29 of them stated they were professionally interested in cyber/mobile security. Figure 5 shows the demographic profile of survey participants.

With the survey, we try to understand user behaviour and mindset while purchasing and using their mobile phones. While 17 people did not provide any answer, most of the others (51 out of 60) bought their smartphones from online markets, technology shops or MNOs. This shows that people mostly trust large sellers when buying their phones. Arguably, this also makes sense from a privacy and security perspective. Large sellers might help users in this regard. For instance, Amazon previously suspended Blu phones which comes with pre-installed spyware [82].

According to the survey results, only 10% use devices which cost less than $100 US dollars. 44 users prefer $351-$700 devices and 27 prefer those costing $701-$1400. We remind that in general there are more security and privacy risks in less expensive smartphones [83].

We also asked questions to learn how long users have been using their phones and how often they change them. Nearly half of the users (40%) stated that their devices were between 2 and 5 years old. Even worse, a remarkable portion (11.7%) have not change their smartphones for at least 5 years. As most vendors support security updates only in their most recent models (two years on average [84]), a significant number of users are at great risk for potential security vulnerabilities. We also asked how often users change their smartphones (this is not asking the previous question again because users may have bought their devices recently). Most users (68 out of 77) change their phones after at least 2 years. As already pointed above, this brings considerable risks. The survey results about the age of smartphones used by participants can be seen in Figure 6.

In order to learn about the criteria in user choices when purchasing smartphones, we asked another question. As expected, price and model are important for most people. Only 14 participants reported they care about privacy and security policies of vendors. 13 users stated that they consider the country of the vendor as part of their purchasing decision. With these results, we argue that users should be informed better about the importance of privacy policies.

We also aim at measuring user knowledge on pre-installed applications and their impacts. We observed that the knowledge of users on the number of pre-installed applications on their device is far from the actual numbers. In Figure 7, we present the number of pre-installed applications users thought they have in their phones. Most of guesses are underestimates (more than half (%55.8) assume only 0-20 pre-installed applications). We note that we calculate the average number of pre-installed applications per device as 294 which is far more than these guesses. In addition, we asked users whether they are informed at any time about pre-installed applications. 31 users (40%) stated that they did not pay any attention to this subject. 27 of them (35%) thought that they were not informed about pre-installed applications.

To understand and compare user behaviour when managing Android permissions, we asked two additional questions. Almost half of them (38 out of 77) stated they checked application permissions before installation. On the other hand, 71% does not bother with periodic regular checks. We remind that even when application permissions are checked by users, permissions given to pre-installed applications cannot be seen.

Do users update applications in their devices when an update is available? Most of them (81%) indicated that they pay attention that applications are up-to-date. However, according to our metadata analysis, as previously mentioned, more than half of the pre-installed applications have not been updated since they came with the devices.

Finally, we asked users if they have heard about regulations like GDPR [48] or KVKK [85] (Personal Data Protection Authority in Turkey). Unfortunately, more than half of the users did not hear any of these regulations. As we discussed, these regulations have an important role for user privacy. Thus, users should be informed better about these regulations and their importance regarding user privacy.

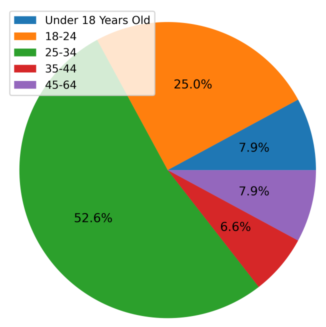
At the end of our survey, we collect email addresses of users to send our analysis results. In our view, users should be notified about pre-installed applications on their phones and their impacts on security and privacy.

## 6 RISK SCORING SYSTEM

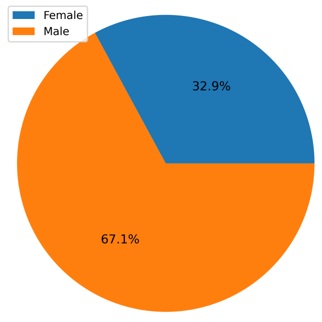
As a result of series of analysis, we obtain various findings regarding pre-installed applications on smart phones that have varying degrees of effects on user security and privacy. However, these findings cannot easily be grasped by an average smart phone user especially when presented and discussed technically. For this reason, we aim at having a scoring system to provide users with information about their devices and pre-installed applications with respect to security and privacy risks in a more clear and concise way. Although, the scores seem inevitably fraught with issues of subjectivity, we believe the end result is still helpful in a certain extent.

While designing our scoring system, we are inspired from the Common Vulnerability Scoring System (CVSS) [86], which assigns scores for vulnerabilities and the quantitative risk assessment methodology for IT systems proposed by Aksu et al. [87], which is built on top of CVSS scores.

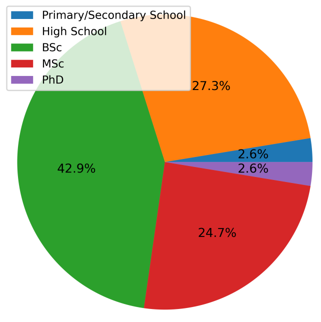
The calculations in the scoring system we present essentially start with the basic risk formula as given in eq. 1.



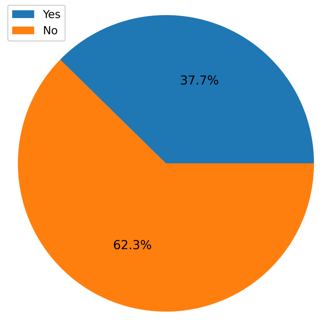
**(a)** Age



**(b)** Gender

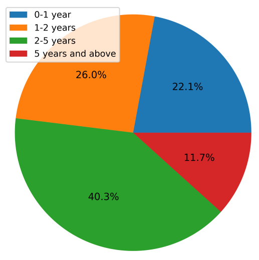


**(c)** Educational background

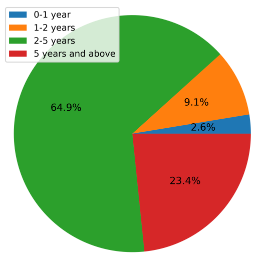


**(d)** Interested in cyber/mobile security?

**Fig. 5** Demographic profile of survey participants.



**(a)** How long have you been using your smartphone?



**(b)** How often do you change your smartphone?

**Fig. 6** Survey results about the age of smartphones.

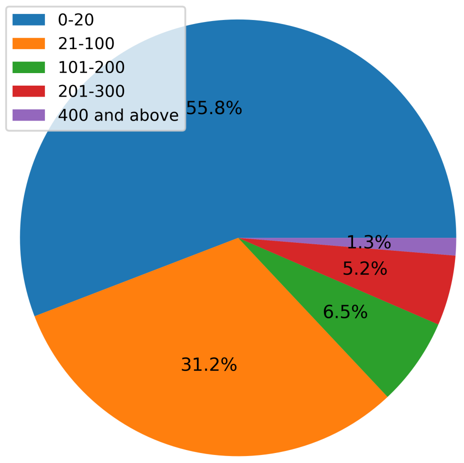


**Fig. 7** How many applications were pre-installed you think when you first bought your phone?

**Table 5** What is the level of difficulty to exploit the finding?

| Criterion | Coefficient ($d_i$) |
|---|---|
| Easy (Almost no requirement) | 1.00 |
| Medium (One of either physical access, an available vulnerability or user interaction is required) | 0.50 |
| Hard (Two of physical access, an available vulnerability and user interaction are required) | 0.25 |
| Very Hard (Physical access, an available vulnerability and user interaction are all required) | 0.10 |

$$Risk = Probability \times Cost \tag{1}$$

With this formula in mind, we first consider each finding separately and calculate a score per finding for each device. Then, we consolidate these scores to obtain a final risk score for each device we analyze.

From this perspective, for each finding we consider, three components contribute to the first parameter (Probability) of the device risk: the number of pre-installed applications ($n_i$) that has the concerned finding $i$, the difficulty level to exploit the concerned finding ($d_i$) and likelihood the user being aware of the exploit (once it happened) ($a_i$). For the later two, we grade each finding according to Tables 5 and 6 where relevant subjective coefficients are determined according to our expertise and experience. To finalize the calculation of the first parameter in the risk formula, we multiply the three components and normalize the result between 0 and 1. We emphasize that the coefficients shown have relative meanings, they do not reflect the absolute values e.g., $d_i = 1.0$ does not mean the concerned exploit is certain.

For the second parameter of the risk, we consult to Figure 7 where subjective coefficients ($I_i$) are available. The first and second parameters are multiplied as shown in eq. 2.

Finally, to obtain consolidated risks per device, we perform one final normalization to the sum to have device scores between 0 and 100. This is captured in eq. 3.

**Table 6** Could the user be aware of the finding and its effects?

| Criterion | Coefficient ($a_i$) |
|---|---|
| The user is unlikely aware of the finding and its effect | 1.00 |
| The user is possibly aware of the finding and its effect | 0.50 |
| The user is likely aware of the finding and its effect | 0.25 |
| The user is most likely aware of the finding and its effect | 0.10 |

**Table 7** What is the level of impact (cost) on user security and privacy?

| Criterion | Coefficient ($I_i$) |
|---|---|
| Affects user privacy or security directly and has very high impact.(Very High) | 1.00 |
| Affects user privacy or security directly and has high impact. (High) | 0.50 |
| Possibly affects user privacy or security with high impact. (Medium) | 0.25 |
| Possibly affects user privacy or security with low impact. (Low) | 0.10 |

$$score_i = Normalize(n_i * d_i * a_i) * I_i \tag{2}$$

$$Total\ Device\ Score = Normalize(\sum_{i=1}^{10} score_i) \tag{3}$$

Below, numerical values assigned to $d_i$, $a_i$ and $I_i$ in our scoring system are given for all of the ten findings, which is split into two groups.

## 6.1 New Findings

The first group contains the findings we analyze and discuss in this work.

**Privileged pre-installed applications.** System user is one of the most privileged users in Android devices and its use by device manufacturers is common. We detect pre-installed applications that run with system user privilege by checking if *sharedUserId* value is *android.uid.system* or not. Even though not directly affecting user privacy and security, unnecessary usage of this privilege definitely opens new attack vectors: $d_1$=0.25 $a_1$=0.50 $I_1$=0.25.

**Applications with *allowBackup* flag enabled.** In Android applications, *allowBackup* flag is used by applications to allow users to backup application data. This feature can be exploited by adversaries to reach application data but only if they have physical access: $d_2$=0.25 $a_2$=0.25 $I_2$=0.25.

**Applications not signed by the manufacturer/vendor.** We examine application certificates and detect the applications not belonging to device manufacturers. These applications mostly do not conform to the security best practices and contain tracker SDKs. Moreover, they are not strictly necessary for the normal operation of the device. When pre-installed, they run with more privileges and permissions as compared to when installed from application markets: $d_3$=0.50 $a_3$=0.50 $I_3$=0.25.

**Applications not updated for more than two years.** In our survey, most users state that they have been using their phones more than two years. Thus, their devices are open to vulnerabilities if pre-installed applications are not updated at least for two years: $d_4$=0.25 $a_4$=0.50 $I_4$=0.10.

**Applications with *usesClearTextTraffic* flag enabled.** One of the best practices in network communication is the use of TLS protocols. After Android API Level 27, applications are not allowed to make cleartext communication unless they set *usesClearTextTraffic* flag as "true" in their manifest file. However this choice is dangerous since network attacks such as man-in-the-middle becomes possible: $d_5$=0.50 $a_5$=0.50 $I_5$=0.25.

**Applications with *debuggable* flag enabled.** Use of this flag in production code is extremely dangerous. In fact, applications with *debuggable* flag set are not allowed to be uploaded to Google Play Store. When this flag is set, application methods and classes can be listed and application behaviour can be manipulated by adversaries having physically access: $d_6$=0.25 $a_6$=0.25 $I_6$=0.50.

**Trackers (excluding crash reporters).** As previously discussed in detail, tracker SDKs that come with pre-installed applications collect various kinds of user data. Users mostly are not aware of them and their activities: $d_7$=1.00 $a_7$=1.00 $I_7$=1.00.

**Vulnerabilities in cloud services.** As discussed earlier, we found a number of vulnerabilities on cloud service configurations used by pre-installed applications. We take into account the difference with respect to the impact of vulnerabilities in Google Maps API and other cloud services: $d_8$=1.00 $a_8$=1.00 $I_8$=0.25 (Google Maps API), $I_8$=1.00 (Others).

## 6.2 Findings from Earlier Research

Our scoring system is enriched further with the results of previous studies. The second group is composed of findings that were analyzed and discussed in previous work (but not considered in a scoring system).

We take advantage of especially one of the most comprehensive study [14] on Android pre-installed applications and include findings regarding dangerous application permissions and exported application components in our scoring system. These findings are analyzed and discussed in the previous work [14]. We use the reported procedure to obtain our results.

**Exported application components not requiring permission(s).** Exported application components can be used by applications to share data and functionality with other applications that are also installed on the device. But, insecure usage of these components may cause various security vulnerabilities. Our analysis on the dataset revealed that many pre-installed applications use exported components without permissions. While being not a direct threat, vulnerabilities in these components still pose a nonnegligible risk for device owners: $d_9$=0.25 $a_9$=0.25 $I_9$=0.10.

**Dangerous permissions.** In Android, dangerous permissions are those which are given to perform actions which may affect user security and privacy. After the Android API Level 23, user consent for the permissions is received at runtime. In theory, this is applied to both third-party and pre-installed applications, however vendors can enable exceptions for pre-installed applications. This can be applied by whitelisting dangerous permissions for specific pre-installed applications [88]. Also, privileged applications which are located in */system* for Android 8.1 and lower, and */system, /product, /vendor* for Android 9.0 and higher can take advantage of privilege permission allowlisting [89]. Moreover, pre-installed apps may expose critical services and data by using custom permissions [14]. This feature allows applications to use runtime permissions without user consent: $d_{10}$=0.25 $a_{10}$=0.25 $I_{10}$=0.25.

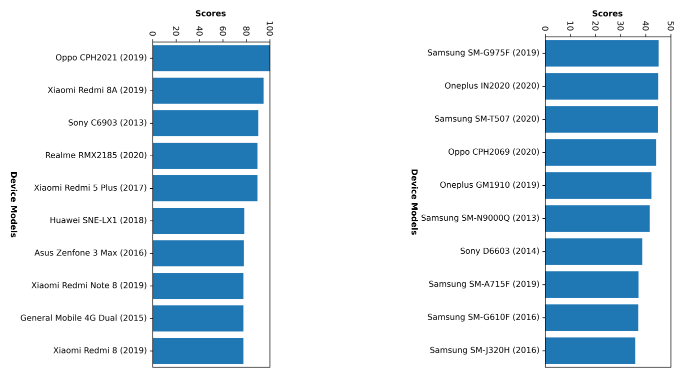
## 6.3 Device Scores

We use 10 different criteria as listed above and eq. 2 and eq. 3 to calculate the final device scores. Devices with the highest scores are the worst with respect to security and privacy impacts of pre-installed applications.

In our analysis, we only consider devices where we could collect more than 50 pre-installed applications since lack of enough data is most likely due to network connection problems. We also do not have sufficient data for some other devices due to various other reasons.

We determine the devices with the highest scores as seen in Figure 8 (a). Sony Xperia Z1 is the device with the highest score in our dataset. Our results also show that 7 of the 10 highest score phones are Samsung devices. Asus and General Mobile devices are also among the devices with the highest scores.

We also determine the best devices with the lowest scores. As seen in Figure 8 (b), most of these devices (6 out of 10) are released in 2019 or later.



**(a)** Devices with the highest-scores.

**(b)** Devices with the lowest scores.

**Fig. 8** Devices with the highest and lowest risk scores.

With a conjecture that the total number of pre-installed applications on a device might be one of the most significant factors in risk scores, we calculate Pearson Correlation and see that the coefficient value is -0.22 which indicates that there is actually a weak negative correlation between the risk scores and the number of pre-installed applications. We illustrate this correlation in Figure 9. We could infer from this figure that risk scores reflect a more complex mix of contributing factors.

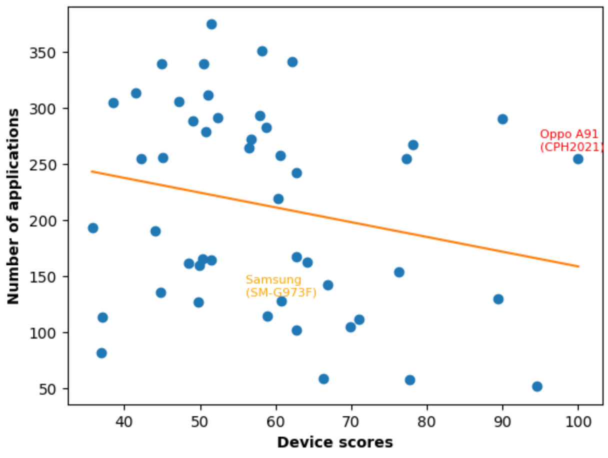


**Fig. 9** The relationship between the number of pre-installed applications on devices and their risk scores.

To conclude this section, we note that with the results of our proposed risk scoring system [17], users could easily compare devices with respect to pre-installed applications and their effects on user security and privacy.

## 7 LIMITATIONS

In this section, we report various limitations of our study as follows:

**Scope**. Our study involves a dataset comprising 14178 apk files and a user study with 77 participants. These numbers are far from being sufficient to draw ultimate results on pre-installed application ecosystem. Our observation is that most people is reluctant on installing an unknown application to their smartphone even when the application is developed for an ethically approved research study. We encourage researchers to use our Android application [15] to conduct follow-up studies to obtain larger datasets of pre-installed applications [2].

**Analysis**. We only analyzed pre-installed applications using static analysis methods, however full functionality of applications cannot be understood only by this method be-

2 On the other hand, for obvious reasons, we do not recommend making it pre-installed.

cause these applications could take advantage of techniques such as reflection, dynamic code loading, native libraries, obfuscation and encryption. Therefore, future work may focus on developing and adopting dynamic analysis methods and platforms for pre-installed applications. Another promising future work could be analyzing privacy policies of pre-installed applications using automation tools [90].

**Scoring System**. The scoring system is designed using the results obtained from a limited number of devices and pre-installed applications. By adding more criteria, the scoring system can be made more comprehensive. We also note that while dangerous permissions may sometimes be actually required by the applications to fulfill their functionality, some others may use these permissions just to access and leak sensitive user data. This distinction can be detected with techniques like taint flow analysis. Additionally, different findings could be merged (e.g., tracker SDKs and dangerous permissions) to improve the reliability of scores.

## 8 CONCLUSION

In this work, we presented a dataset made publicly available for Android pre-installed applications. We analyzed pre-installed applications in various aspects and developed a scoring system, grading and consolidating the effects on user security and privacy. We also conducted a user study to understand and measure the knowledge and perceptions of users about pre-installed applications and their activities.

In our tracker SDK analysis, we observed that most of the tracker SDKs exist in third-party applications. However, users cannot uninstall these applications, they could only deactivate them. Although these applications are not required for proper device operation, they have serious security and usability impacts. Also, we detected tracker SDKs on vendor pre-installed applications, which confirms that vendors and third-party firms collaborate with each other.

We analyzed critical manifest file attributes and flags such as *sharedUserId, allowBackup, debuggable, usesClearTextTraffic* and determined various pre-installed applications having critical security vulnerabilities. Vendors are urged to follow security best practices while developing pre-installed applications.

We examined cloud services in pre-installed applications and detected various vulnerabilities that affect user security and privacy in varying levels. Some of these allow even unauthorized access to data of other applications and users.

The user survey we conducted to learn knowledge and perception of users about pre-installed applications showed that most of the participants have limited knowledge about them. One takeaway is that users should be informed better about pre-installed applications and their effects on user security and privacy. For this purpose, we developed a website [17] and published our analysis results for each device we analyzed.

The scoring system we developed takes into account the difficulty of exploiting, the awareness level of users and the impact on security and privacy. We evaluated the devices with respect to ten different findings and the normalized sum of scores for findings gave us a total device score. With this score, users may easily form an opinion concerning the security and privacy impacts of mobile devices and pre-installed applications.

To sum up, pre-installed applications in Android devices can affect security and privacy of users in multiple ways. However, this topic has not drawn much attention in academic literature. We encourage researchers to take advantage of our available dataset [16]. We believe there are still many aspects of pre-installed applications awaiting to be uncovered.

## COMPLIANCE WITH ETHICAL STANDARDS

Data collection and user survey that are made as part of this study is ethically approved by TOBB University of Economics & Technology Human Research Evaluation Board [42]. Collected information does not contain any personal data and is not shared with any third party.

## COMPETING INTERESTS

The authors declare that they have no competing interests.

## RESEARCH DATA POLICY AND DATA AVAILABILITY STATEMENT

The data set is created as part of this study can be accessed from Kaggle [16]. Also, analyses results are available in our website [17].

## REFERENCES

[1] “Mobile operating system market share worldwide | statcounter global stats,” https://gs.statcounter.com/os-market-share/mobile/worldwide, (Accessed on 29/11/2022).

[2] “Android open source project,” https://source.android.com/, (Accessed on 29/11/2022).

[3] “Android compatibility program overview - android open source project,” https://source.android.com/compatibility/overview?hl=en, (Accessed on 29/11/2022).

[4] “Android compatibility definition document,” https://source.android.com/compatibility/cdd, (Accessed on 29/11/2022).

[5] “Compatibility test suite - android open source project,” https://source.android.com/compatibility/cts, (Accessed on 29/11/2022).

[6] “Android - certified,” https://www.android.com/certified/, (Accessed on 29/11/2022).

[7] “Android certified partners,” https://www.android.com/certified/partners/, (Accessed on 29/11/2022).

[8] “Securing the system: A deep dive into reversing android preinstalled apps,” https://i.blackhat.com/USA-19/Thursday/us-19-Stone-Securing-The-System-A-Deep-Dive-Into-Reversing-Android-Preinstalled-Apps.pdf, (Accessed on 29/11/2022).

[9] “Android firmware sending private information without consent - kryptowire,” https://www.kryptowire.com/kryptowire-discovers-mobile-phone-firmware-transmitted-personally-identifiable-information-pii-without-user-consent-disclosure/, (Accessed on 29/11/2022).

[10] “Google android security 2018 report final.pdf,” https://source.android.com/security/reports/Google_Android_Security_2018_Report_Final.pdf, (Accessed on 29/11/2022).

[11] “Two weeks of securing samsung devices: Part 1 - oversecured blog,” https://blog.oversecured.com/Two-weeks-of-securing-Samsung-devices-Part-1/, (Accessed on 29/11/2022).

[12] “Facebook app can’t be deleted from certain samsung phones - bloomberg,” https://www.bloomberg.com/news/articles/2019-01-08/samsung-phone-users-get-a-shock-they-can-t-delete-facebook, (Accessed on 29/11/2022).

[13] “Facebook gave device makers deep access to data on users and friends - the new york times,” https://www.nytimes.com/interactive/2018/06/03/technology/facebook-device-partners-users-friends-data.html?mtrref=undefined&gwh=DFAE7B3996870E0D2452CDBF4B2F1154&gwt=pay&assetType=PAYWALL, (Accessed on 29/11/2022).

[14] J. Gamba, M. Rashed, A. Razaghpanah, J. Tapiador, and N. Vallina-Rodriguez, “An analysis of pre-installed android software,” in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, pp. 1039–1055.

[15] “Pre-app collector application - google play,” https://play.google.com/store/apps/details?id=com.preappcollector, (Accessed on 29/11/2022).

[16] “Android pre-installed applications | kaggle,” https://www.kaggle.com/abdullahzbay/android-preinstalled-applications, (Accessed on 29/11/2022).

[17] “Pre-app collector website,” https://preappcollector.com/, (Accessed on 29/11/2022).

[18] “Google play store,” https://play.google.com/store, (Accessed on 29/11/2022).

[19] “Galaxy store apps - the official samsung galaxy site,” https://www.samsung.com/global/galaxy/apps/galaxy-store/, (Accessed on 29/11/2022).

[20] “The amazon app,” https://www.amazon.com/gp/mas/get/amazonapp, (Accessed on 29/11/2022).

[21] “F-droid - free and open source android app repository,” https://f-droid.org/en/, (Accessed on 29/11/2022).

[22] “Apkpure.com,” https://apkpure.com/, (Accessed on 29/11/2022).

[23] K. W. Y. Au, Y. F. Zhou, Z. Huang, and D. Lie, “Pscout: Analyzing the android permission specification,” in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, ser. CCS ’12. New York, NY, USA: Association for Computing Machinery, 2012, pp. 217–228. [Online]. Available: https://doi.org/10.1145/2382196.2382222

[24] A. P. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner, “Android permissions demystified,” in *Proceedings of the 18th ACM Conference on Computer and Communications Security*, ser. CCS ’11. New York, NY, USA: Association for Computing Machinery, 2011, pp. 627–638. [Online]. Available: https://doi.org/10.1145/2046707.2046779

[25] C. Gibler, J. Crussell, J. Erickson, and H. Chen, “Androidleaks: Automatically detecting potential privacy leaks in android applications on a large scale,” in *Trust and Trustworthy Computing*, S. Katzenbeisser, E. Weippl, L. J. Camp, M. Volkamer, M. Reiter, and X. Zhang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 291–307.

[26] G. Tuncay, S. Demetriou, K. Ganju, and C. Gunter, “Resolving the predicament of android custom permissions,” 01 2018.

[27] B. Liu, J. Lin, and N. Sadeh, “Reconciling mobile app privacy and usability on smartphones: Could user privacy profiles help?” in *Proceedings of the 23rd International Conference on World Wide Web*, ser. WWW ’14. New York, NY, USA: Association for Computing Machinery, 2014, pp. 201–212. [Online]. Available: https://doi.org/10.1145/2566486.2568035

[28] A. Razaghpanah, R. Nithyanand, N. Vallina-Rodriguez, S. Sundaresan, M. Allman, C. Kreibich, and P. Gill, “Apps, trackers, privacy, and regulators: A global study of the mobile tracking ecosystem,” in *NDSS*, 2018.

[29] R. Binns, U. Lyngs, M. Van Kleek, J. Zhao, T. Libert, and N. Shadbolt, “Third party tracking in the mobile ecosystem,” in *Proceedings of the 10th ACM Conference on Web Science*, ser. WebSci ’18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 23–31. [Online]. Available: https://doi.org/10.1145/3201064.3201089

[30] H. Wang, H. Li, and Y. Guo, “Understanding the evolution of mobile app ecosystems: A longitudinal measurement study of google play,” in *The World Wide Web Conference*, ser. WWW ’19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1988–1999. [Online]. Available: https://doi.org/10.1145/3308558.3313611

[31] B. Hu, Q. Lin, Y. Zheng, Q. Yan, M. Troglia, and Q. Wang, “Characterizing location-based mobile tracking in mobile ad networks,” in *2019 IEEE Conference on Communications and Network Security (CNS)*, 2019, pp. 223–231.

[32] “Unsecured cloud configurations exposing information in thousands of mobile apps,” https://blog.zimperium.com/unsecured-cloud-configurations-exposing-information-in-thousands-of-mobile-apps/, (Accessed on 29/11/2021).

[33] “Mobile app developers’ misconfiguration of third party services leave personal data of over 100 million exposed - check point research,” https://research.checkpoint.com/2021/mobile-app-developers-misconfiguration-of-third-party-services-leave-personal-data-of-over-100-million-exposed/, (Accessed on 29/11/2021).

[34] “Mobile security testing guide,” https://mobile-security.gitbook.io/mobile-security-testing-guide/, (Accessed on 29/11/2021).

[35] D. Barrera, J. Clark, D. McCarney, and P. C. van Oorschot, “Understanding and improving app installation security mechanisms through empirical analysis of android,” in *Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, ser. SPSM ’12. New York, NY, USA: Association for Computing Machinery, 2012, pp. 81–92. [Online]. Available: https://doi.org/10.1145/2381934.2381949

[36] E. Ratazzi, Y. Aafer, A. Ahlawat, H. Hao, Y. Wang, and W. Du, “A systematic security evaluation of android’s multi-user framework,” *ArXiv*, vol. abs/1410.7752, 2014.

[37] S. M. Dye and K. Scarfone, “A standard for developing secure mobile applications,” *Computer Standards & Interfaces*, vol. 36, no. 3, pp. 524–530, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548913001268

[38] H. Elahi, G. Wang, and J. Chen, “Pleasure or pain? an evaluation of the costs and utilities of bloatware applications in android smartphones,” *J. Netw. Comput. Appl.*, vol. 157, no. C, May 2020. [Online]. Available: https://doi.org/10.1016/j.jnca.2020.102578

[39] M. Elsabagh, R. Johnson, A. Stavrou, C. Zuo, Q. Zhao, and Z. Lin, “FIRMSCOPE: Automatic uncovering of privilege-escalation vulnerabilities in pre-installed apps in android firmware,” in *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, Aug. 2020, pp. 2379–2396. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/elsabagh

[40] E. Blázquez, S. Pastrana, A. Feal, J. Gamba, P. Kotzias, N. Vallina-Rodriguez, and J. Tapiador, “Trouble over-the-air: An analysis of fota apps in the android ecosystem,” in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 1606–1622.

[41] A. Ozbay and K. Bicakci, “Android pre-installed applications effects on user’s privacy,” in *2021 International Conference on Information Security and Cryptology (ISCTURKEY)*, 2021, pp. 12–17.

[42] “ethical_approval.pdf,” https://preappcollector.com/static/ethical_approval.pdf, (Accessed on 29/11/2022).

[43] “Androguard,” https://github.com/androguard/androguard, (Accessed on 29/11/2022).

[44] “Application signing,” https://developer.android.com/studio/publish/app-signing, (Accessed on 29/11/2022).

[45] “exodus,” https://reports.exodus-privacy.eu.org/en/, (Accessed on 29/11/2022).

[46] “exodus-standalone,” https://github.com/Exodus-Privacy/exodus-standalone, (Accessed on 29/11/2022).

[47] “Google advertising id - play console help,” https://support.google.com/googleplay/android-developer/answer/6048248, (Accessed on 29/11/2022).

[48] “General data protection regulation (gdpr),” https://gdpr-info.eu/, (Accessed on 29/11/2022).

[49] “California consumer privacy act (ccpa),” https://oag.ca.gov/privacy/ccpa, (Accessed on 29/11/2022).

[50] “Crunchbase: Discover innovative companies and the people behind them,” https://www.crunchbase.com/, (Accessed on 29/11/2022).

[51] "Why do you even need the imei?" https://blog.appcensus.io/2019/04/26/why-do-you-even-need-the-imei/, (Accessed on 29/11/2022).

[52] "Exclusive: Warning over chinese mobile giant xiaomi recording millions of people's 'private' web and phone use," https://www.forbes.com/sites/thomasbrewster/2020/04/30/exclusive-warning-over-chinese-mobile-giant-xiaomi-recording-millions-of-peoples-private-web-and-phone-use/?sh=7579d95c1b2a, (Accessed on 29/11/2022).

[53] "Baidu's and don'ts: Privacy and security issues in baidu browser," https://citizenlab.ca/2016/02/privacy-security-issues-baidu-browser/, (Accessed on 29/11/2022).

[54] "Data leakage found from android apps on google play with millions of downloads," https://unit42.paloaltonetworks.com/android-apps-data-leakage/, (Accessed on 29/11/2022).

[55] "Dji releases security findings it hopes will quash 'chinese spying' fears," https://gizmodo.com/dji-releases-security-findings-it-hopes-will-quash-chin-1825469976, (Accessed on 29/11/2022).

[56] "Privacy policy - mintegral," https://www.mintegral.com/en/privacy/, (Accessed on 29/11/2022).

[57] "Privacy policy - moengage," https://www.moengage.com/privacy-policy/, (Accessed on 29/11/2022).

[58] "Report: Aurora mobile's jpush sdk - the appcensus blog," https://blog.appcensus.io/2020/09/15/report-aurora-mobiles-jpush-sdk/, (Accessed on 29/11/2022).

[59] "Industry collaborations - mopub," https://www.mopub.com/en, (Accessed on 29/11/2022).

[60] Z. Wang, "Systematic government access to private-sector data in China," *International Data Privacy Law*, vol. 2, no. 4, pp. 220–229, 07 2012. [Online]. Available: https://doi.org/10.1093/idpl/ips017

[61] "App manifest overview | android developers," https://developer.android.com/guide/topics/manifest/manifest-intro, (Accessed on 29/11/2022).

[62] "Nvd - cve-2018-14825," https://nvd.nist.gov/vuln/detail/CVE-2018-14825, (Accessed on 29/11/2022).

[63] "Android debug bridge (adb) | android developers," https://developer.android.com/studio/command-line/adb, (Accessed on 29/11/2022).

[64] "Android developers blog: Protecting against unintentional regressions to cleartext traffic in your android apps," https://android-developers.googleblog.com/2016/04/protecting-against-unintentional.html, (Accessed on 29/11/2022).

[65] "jdb - the java debugger," https://docs.oracle.com/javase/7/docs/technotes/tools/windows/jdb.html, (Accessed on 29/11/2022).

[66] "Firebase," https://firebase.google.com/, (Accessed on 29/11/2022).

[67] "Amazon web services (aws) - cloud computing services," https://aws.amazon.com/, (Accessed on 29/11/2022).

[68] "Cloud computing services | microsoft azure," https://azure.microsoft.com/en-us/, (Accessed on 29/11/2022).

[69] "Google maps platform | google developers," https://developers.google.com/maps, (Accessed on 29/11/2022).

[70] "dwisiswant0/apkleaks: Scanning apk file for uris, endpoints & secrets." https://github.com/dwisiswant0/apkleaks, (Accessed on 29/11/2022).

[71] "skylot/jadx: Dex to java decompiler," https://github.com/skylot/jadx, (Accessed on 29/11/2022).

[72] "Apktool - a tool for reverse engineering 3rd party, closed, binary android apps." https://ibotpeaches.github.io/Apktool/, (Accessed on 29/11/2022).

[73] "gmapsapiscanner," https://github.com/ozguralp/gmapsapiscanner, (Accessed on 29/11/2022).

[74] "Sending messages using incoming webhooks | slack," https://api.slack.com/messaging/webhooks, (Accessed on 29/11/2022).

[75] "Oauth 2.0 - oauth," https://oauth.net/2/, (Accessed on 29/11/2022).

[76] "Billing: Mapping previous skus to new skus | google maps platform," https://developers.google.com/maps/billing/sku-mapping-old-to-new, (Accessed on 29/11/2022).

[77] "Google maps platform billing | google developers," https://developers.google.com/maps/billing/gmp-billing, (Accessed on 29/11/2022).

[78] "Global cloud infrastructure market share 2021 | statista," https://www.statista.com/statistics/967365/worldwide-cloud-infrastructure-services-market-share-vendor/, (Accessed on 29/11/2022).

[79] "Aws general reference - reference guide," https://docs.aws.amazon.com/general/latest/gr/aws-general.pdf#aws-access-keys-best-practices, (Accessed on 29/11/2022).

[80] “Authenticating users of aws mobile applications with a token vending machine - aws articles,” https://aws.amazon.com/tr/articles/authenticating-users-of-aws-mobile-applications-with-a-token-vending-machine/, (Accessed on 29/11/2022).

[81] “Protecting mobile apps with pkce - oauth 2.0 simplified,” https://www.oauth.com/oauth2-servers/pkce/, (Accessed on 29/11/2022).

[82] “Amazon suspends sales of blu phones for including preloaded spyware, again - the verge,” https://www.theverge.com/2017/7/31/16072786/amazon-blu-suspended-android-spyware-user-data-theft, (Accessed on 29/11/2022).

[83] “Buying a smart phone on the cheap? privacy might be the price you have to pay - privacy international,” https://privacyinternational.org/long-read/3226/buying-smart-phone-cheap-privacy-might-be-price-you-have-pay, (Accessed on 29/11/2022).

[84] “Mobile security updates: Understanding the issues,” https://www.ftc.gov/system/files/documents/reports/mobile-security-updates-understanding-issues/mobile_security_updates_understanding_the_issues_publication_final.pdf, (Accessed on 29/11/2022).

[85] “Kişisel verileri koruma kurumu | kvkk | personal data protection authority,” https://www.kvkk.gov.tr/en/, (Accessed on 29/11/2022).

[86] “What are cvss scores | balbix,” https://www.balbix.com/insights/understanding-cvss-scores/, (Accessed on 29/11/2022).

[87] M. U. Aksu, M. H. Dilek, E. I. Tatli, K. Bicakci, H. I. Dirik, M. U. Demirezen, and T. Aykir, “A quantitative cvss-based cyber security risk assessment methodology for it systems,” in *2017 International Carnahan Conference on Security Technology (ICCST)*. IEEE, 2017, pp. 1–8.

[88] “Runtime permissions | android open source project,” https://source.android.com/devices/tech/config/runtime_perms?hl=en#creating-exceptions, (Accessed on 29/11/2022).

[89] “Privileged permission allowlisting | android open source project,” https://source.android.com/devices/tech/config/perms-allowlist?hl=en, (Accessed on 29/11/2022).

[90] H. Harkous, K. Fawaz, R. Lebret, F. Schaub, K. G. Shin, and K. Aberer, “Polisis: Automated analysis and presentation of privacy policies using deep learning,” in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 531–548.

## APPENDIX A Survey Questions

1) Please select your age range.
a. Under 18 years old
b. 18-24
c. 25-34
d. 35-44
e. 45-64

2) Please select your gender.
a. Female
b. Male
c. Prefer not to answer

3) What is your educational background?
a. Primary School-Secondary School
b. High school
c. Bachelor (BSc)
d. Master of Science (MSc)
e. Doctorate (PhD)

4) Are you professionally interested in cyber security / mobile security?
a. Yes
b. No

5) Where did you buy your smartphone?
Technology Store, Telecommunication Company, Local Store, 2nd Hand Seller, Online Market etc. (Text Box)

6) How much money did you pay for your smartphone?
a. 0-130 $
b. 131-350 $
c. 351-700 $
d. 701-1400 $
e. 1400 $ and above

7) How long have you been using your smartphone?
a. 0-1 Year
b. 1-2 Years
c. 2-5 Years
d. 5 Years and above

8) How often do you change your smartphone?
a. 0-1 Year
b. 1-2 Years
c. 2-5 Years
d. 5 Years and above

9) How many pre-installed applications (already installed on the device when the device came out of the box) do you think there are when you first bought your phone?
a. 0-20
b. 21-100
c. 101-200
d. 201-300
e. 301-400
f. 400 and above

10) When purchasing a smartphone, select the factors that affect your purchasing decision. (Note: Users can choose multiple choices)
a. Price
b. Model
c. Popularity
d. Country of manufacturer (Samsung: South Korea, Huawei: China, etc.)
e. Security and Privacy Policy of the Manufacturer / Seller
f. Whether it is Sold / Manufactured by large and well-known companies

11) While setting up your smartphone, have you been informed about the pre-installed applications and the operations these applications perform and the data they collect?
a. Yes, I have been informed.
b. No, I havent́ been informed.
c. I did not pay attention / I did not read.

12) Did you give any permission for pre-installed apps?
a. No.
b. Yes.
c. I did not pay attention / I did not recall.

13) Do you pay attention to what permissions the apps you install on your phone use?
a. No, I don’t pay attention.
b. Yes, I pay attention.

14) Do you regularly check these permissions?
a. Yes, I’m checking.
b. No, I’m not checking.

15) Do you check that applications on your smartphone are up-to-date?
a. Yes
b. No

16) Do you think you know enough about General Data Protection Regulation (GDPR) or Personal Data Protection Authority in Turkey (KVKK)?
a. Yes
b. No

ITU/JWCC

# A Survey On Security and Privacy Aspects and Solutions for Federated Learning in Mobile Communication Networks

**Şükrü Erdal**[1], **Ferhat Karakoç**[2], **and Enver Özdemir**[1]

[1] Istanbul Technical University, Informatics Institute, Istanbul, 34469, Turkey
[2] Ericsson Research, Arı Teknokent 2, Istanbul, 34485, Turkey

**Abstract:** In this study, we delve into cutting-edge solutions for security-centric, privacy-enhanced federated learning, a rapidly evolving area of research that bridges the gap between data privacy and collaborative machine learning. Our analysis offers a comprehensive comparative evaluation of existing methodologies, shedding light on the strengths and limitations of current approaches. By introducing new perspectives, we aim to push the boundaries of secure federated learning, exploring techniques that enhance data protection without compromising learning efficiency. Additionally, we highlight emerging challenges and opportunities in the field, emphasizing the importance of scalable, privacy-preserving mechanisms in decentralized systems. As federated learning continues to gain traction across various sectors such as healthcare, finance, and IoT, our study serves as a foundation for future research, identifying key areas for innovation and improvement. This forward-looking approach ensures that federated learning can continue to evolve as a trustworthy and robust solution for privacy-sensitive applications, addressing both current and future security concerns.

**Keywords:** Federated learning, privacy, secure aggregation, homomorphic encryption, secure multi-party computation.

## Mobil İletişim Ağları Alanında Güvenli ve Mahremiyet Odaklı Federe Öğrenme Üzerine Bir Araştırma

**Özet:** Bu çalışmada, veri gizliliği ile iş birliğine dayalı makine öğrenimi arasındaki boşluğu dolduran güvenlik odaklı, gizlilik artırılmış federe öğrenme için en son çözümleri ele alıyoruz. Analizimiz, mevcut metodolojilerin kapsamlı bir karşılaştırmalı değerlendirmesini sunarak, mevcut yaklaşımların güçlü ve zayıf yönlerine ışık tutuyor. Yeni bakış açıları sunarak, güvenli federe öğrenmenin sınırlarını zorlamayı ve öğrenme verimliliğinden ödün vermeden veri korumasını artıran teknikleri keşfetmeyi amaçlıyoruz. Ayrıca, merkezi olmayan sistemlerde ölçeklenebilir, mahremiyet odaklı mekanizmaların önemini vurgulayan ortaya çıkan zorluklar ve fırsatlara dikkat çekiyoruz. Federe öğrenme, sağlık hizmetleri, finans ve Nesnelerin Interneti gibi çeşitli sektörlerde hız kazanmaya devam ederken, çalışmamız, yenilik ve gelişim için önemli alanları belirleyerek, gelecekteki araştırmalar için bir temel niteliği taşıyor. Bu ileriye dönük yaklaşım, federe öğrenmenin hem mevcut hem de gelecekteki güvenlik kaygılarını ele alarak, gizliliğe duyarlı uygulamalar için güvenilir ve sağlam bir çözüm olarak gelişmeye devam etmesini sağlıyor.

**Anahtar Kelimeler:** Federe öğrenme, mahremiyet, güvenli birleştirme, homomorfik şifreleme, çok taraflı güvenli hesaplama.

REVIEW PAPER

Corresponding Author: Şükrü Erdal, erdal21@itu.edu.tr

Reference: Ş. Erdal, F. Karakoç, and E.Özdemir, (2024), "A Survey On Security and Privacy Aspects and Solutions for Federated Learning in Mobile Communication Networks," *ITU Journ. Wireless Comm. Cyber.*, 1, (1) 29–40.

Submission Date: May, 12, 2024
Acceptance Date: June, 2, 2024
Online Publishing: Sept, 30, 2024

## 1 INTRODUCTION

Machine Learning (ML) is garnering increased attention due to its promising benefits across a broad spectrum of applications, fulfilling needs such as automated management, optimization, enhanced user experiences, and security automation in IoT and mobile communication use cases. This surge in interest has led to a significant uptick in ML deployment. Given that decisions driven by ML can carry substantial implications, the security of ML systems is paramount. Moreover, considering ML's inherent reliance on vast datasets, which may include sensitive corporate information or personal data, concerns regarding privacy and data security are inevitable. To enhance the precision of ML models, incorporating data from multiple providers—a practice known as collaborative ML—is often preferred. However, the involvement of numerous participants in collaborative ML setups amplifies the risks associated with privacy and security [1]. A notable approach to privacy-conscious collaborative ML is federated learning (FL) [2]. This framework consists of a central server and multiple clients. The server initiates the process by distributing an initial global model to the clients. Subsequently, the clients utilize their private datasets to refine the global model locally. These enhanced models are then transmitted back to the server. Upon receipt, the server aggregates the updates to produce an improved iteration of the global model. This cycle is reiterated until the global model reaches convergence. Nonetheless, the standard implementation of FL falls short in addressing privacy issues, as the model updates relayed from clients to the server may inadvertently disclose sensitive information from the private training datasets of the clients.

The literature abounds with studies aimed at tackling the security (e.g., poisoning attacks) and privacy challenges inherent in collaborative FL [3]. For instance, a prevalent strategy to bolster privacy is the implementation of secure aggregation protocols. These protocols ensure that individual model updates are transmitted to the server in a manner that safeguards the confidentiality of the data. An additional concern pertains to the security of the FL process. Given the multitude of participants within the FL framework, there is a plausible risk that one or more clients may become compromised. Such entities might attempt to engage in malicious activities, including manipulating the global model to serve nefarious objectives—a type of assault known as a poisoning attack [4]. To mitigate such attacks, the server must implement anomaly detection mechanisms to scrutinize model updates from clients. However, if a secure aggregation protocol is in place, the server's ability to inspect individual model updates for poisoning attack detection is hindered due to the encryption. Consequently, the principal challenge lies in achieving both security and privacy concurrently. While the literature presents several solutions that tackle this issue, they often exhibit limitations regarding their practical applicability.

In this paper, we examine the existing privacy and security-enhanced FL solutions, categorize them considering generic approaches used, compare them in terms of their advantages and drawbacks, and analyze their applicability in the industrial IoT (IIoT) domain which is one of the promising domains where the ML is widely used. To be able to analyze the applicability of these privacy and security-enhanced solutions in this domain, we first provide information about the requirements of the application of ML in mobile communication networks, we then conduct a comprehensive survey of existing FL solutions tailored for mobile communication networks. Upon reviewing these FL solutions, we present an overview of generic security and privacy-enhanced FL strategies found in the literature and evaluate their relevance and effectiveness within this specific context. Despite the abundance of surveys on FL, we recognize a gap in the scholarly discourse; to our knowledge, no existing survey thoroughly examines the application and challenges of FL within the mobile communication domain.

The paper is organized as follows. In Section 2, we mention the existing surveys and highlight our surveys. Section 3 is devoted to domain-specific information and analysis of existing FL solutions applied in this domain. We revisit the generic existing solutions for privacy and security-enhanced FL in the literature in Section 4. We share some identified possible research directions and conclude the paper with Section 5.

## 2 SURVEYS ON FEDERATED LEARNING

In this section, we commence by aggregating existing surveys that focus on security and privacy in FL. Subsequently, we furnish insights into the surveys that delve into the application of FL in the mobile communication sphere. Lastly, we underscore the unique contributions of our study.

### 2.1 Security and Privacy in Federated Learning

There have been some survey studies that present the state of the art on security and/or privacy in FL. This subsection gives information about related existing surveys and compares them in Table 1 in terms of covering security aspects ("Security"), privacy aspects ("Privacy"), mitigation solutions "Defense", security and privacy simultaneously ("S&P"), and mobile communication domain ("Mobile Comm").

Soykan et al. [1] presented generic security and privacy attacks not specific to FL but for collaborative ML in general. The work also pointed out generic solutions such as confidential computing, secure multi-party computation, homomorphic encryption, and differential privacy. In 2021, Mothukuri et al. focused on security and privacy issues specific to FL [5]. They first provide an overview and clas-

**Table 1** Comparison of the surveys

| Survey | Security | Privacy | Defense | S&P | Mobile Comm |
|---|---|---|---|---|---|
| Soykan et al. [1] | ✓ | ✓ | ✓ | | |
| Mothukuri et al. [5] | ✓ | ✓ | ✓ | | |
| Blanco-Justicia et al. [6] | ✓ | ✓ | ✓ | | |
| Truong et al. [7] | | ✓ | ✓ | | |
| Bouacida and Mohapatra [8] | ✓ | | ✓ | | |
| Mansouri et al. [9] | ✓ | ✓ | ✓ | ✓ | |
| Enthoven and Al-Ars [10] | | ✓ | ✓ | | |
| Lyu et al. [11] | ✓ | ✓ | ✓ | | |
| Mao et al. [12] | ✓ | ✓ | ✓ | | |
| Asad et al. [13] | | ✓ | ✓ | | |
| Akhtarshenas et al. [14] | ✓ | ✓ | ✓ | | |
| Our survey | ✓ | ✓ | ✓ | ✓ | ✓ |

sification of FL approaches, and then identify and analyze security and privacy threats in FL settings, mitigation techniques, and trade-off costs, and share information about existing defense mechanisms and possible future directions. Another work presented in 2021 also presents security and privacy challenges in FL and also proposes a solution to enhance privacy by breaking the link between the local model update owners and local models [6]. To achieve unlinkability, they proposed to use peer-to-peer decentralized networks for anonymous communication channels. One existing study looks at the problem from the regulation perspective [7]. Since they focused on privacy regulations, they focused only on privacy aspects in FL, mainly considering privacy requirements from GDPR perspective. They analyze challenges taking the GDPR guidelines into account, which results in the need to use strong cryptographic tools. The study presented by Bouacida and Mohapatra investigated vulnerabilities in FL and performed a systematical classification of the threats in FL [8]. Beyond the research concentrating on security and privacy threats, a select number of surveys extend their scope to encompass solutions for secure and privacy-enhanced FL. The study of Mansouri et al. focused only on secure aggregation protocols which is a widely adapted privacy-enhancing technology to protect privacy in FL [9]. They classified the secure aggregation protocols and presented a comparison of the solutions. Enthoven et al. gave an overview of privacy attacks in FL and surveyed mitigation methods [10]. They also focused on insider attacks while identifying threats, and categorized the threats considering attacker types and capabilities. In a recent survey by Lyu et al., the authors presented an overview of privacy and robustness in FL [11]. Robustness means being able to secure against security attacks. They also identified defense mechanisms against certain types of attacks. They provided helpful guidance for the robust and privacy-enhanced FL. Another survey by Mao et al. discusses security and privacy concerns in FL, mentions some privacy-preserving technologies, and discusses possible future works [12]. Asad et al. examine the challenges related to communication limitations, resource allocation, client selection, and optimization methods in FL [13]. That survey underscores the importance of mitigating communication expenses to enhance the efficiency and scalability of FL. It also outlines future pathways for FL concerning communication costs. Various FL structures are examined in Akhtarshenas et al.'s study by evaluating their efficiency, accuracy, and privacy aspects [14]. The research scrutinizes contributions and findings focusing on security, resource optimization, and FL applications across different domains.

## 2.2 Surveys on Application of Federated Learning in Mobile Communication

Another type of survey we searched is on the application of FL in mobile communication systems. The survey [15] by Sirohi et al. gives a comprehensive picture of FL and its application in underwater, ground, air, and space environments considering security and privacy challenges. Although it gives detailed and organized information about research around FL, they don't touch the details and comparison of the solutions that try to address the security and privacy aspects in FL simultaneously. Al-Quraan, Mohjazi, et al. [16] focus on potential research directions to extend FL's capabilities into emerging areas such as 5G and 6G wireless communication systems. The specific requirements of 6G communication and the fundamental challenges faced by FL in the context of 6G applications are examined in Liu, Yuan, et al. 's study [17]. To address these challenges, a detailed overview of emerging advanced FL methods is tailored for 6G communications. These methods include communication-efficient FL, secure FL, and effective FL approaches. In a recent survey paper, Rahman et al. [18] discuss the integration of FL into the Information-Centric Networking (ICT) in the IoT domain.

Zuo et al. [19] provide a review of the recent advancements in utilizing blockchain and AI for 6G wireless communications. It thoroughly explores the integration possibilities of blockchain and AI, and motivations for integrating these technologies into 6G wireless communications, followed by discussions on their simultaneous deployment in secure services and IoT smart applications. Specifically, the survey delves into secure services supported by blockchain and AI, such as spectrum management, computation allocation, content caching, and security and privacy services. Overall, the survey aims to comprehensively explore the potential of blockchain and AI technologies in enhancing wireless communications for 6G networks.

Abimannan et al. [20] emphasize the importance of FL and multi-access edge computing (MEC) in air quality monitoring and forecasting, particularly within smart environments and cities. It highlights the increasing interest and emerging trends, as well as the potential benefits and constraints of these technologies enhanced with deep ML. With the integration of new wireless mobile networks (5G and beyond) with MEC, several applications, including air quality monitoring and control, stand to benefit from improved connectivity, faster processing, and enhanced real-time analytics enabled by FL. FL facilitates collaborative model training across edge devices, ensuring data privacy and security while handling heterogeneous data from diverse sources and IoT devices at the network edge. Future research directions include addressing limitations and maximizing the potential of FL and MEC in building effective air quality monitoring and decision-making ecosystems to protect public health.

Khalek et al. [21] investigate the role of ML-driven Cognitive Radio (CR) in various network domains, including IoT, mobile, vehicular, railway, and UAV networks. Across each network category, the paper delved into the motivations behind adopting ML-driven CR and conducted a comprehensive analysis of recent research trends.

Driss et al. [22] examine FL's role within wireless communication networks. Additionally, it reviews recent contributions utilizing FL to enhance communication and Key Performance Indicators (KPIs) within the protocol stack. Lastly, the paper discusses insights and challenges associated with deploying FL strategies in 5G, 6G, and beyond.

Ferrag et al. 's survey [23] is an expository paper on the state-of-the-art vulnerabilities and defenses in FL for 6G-enabled IoT systems. Also, the paper synthesized existing research on ML security for 6G-IoT, categorizing threats across centralized, federated, and distributed learning modes. Through research and analysis, eight categories of threat models against ML have been identified: backdoor attacks, adversarial examples, combined attacks, poisoning attacks, Sybil attacks, Byzantine attacks, inference attacks, and dropping attacks. Additionally, the survey reviewed current defense methods against vulnerabilities in FL.

I. Bartsiokas et al. 's survey [24] focuses on deploying FL-based approaches within different Physical Layer (PHY) sub-problems in 6G wireless networks. To demonstrate the effectiveness of FL-based schemes in the PHY domain, simulations are conducted to investigate the problem of Relay Node (RN) placement in 6G networks. Two schemes are compared, one employing Centralized Learning (CL) and the other FL.

The complexity of data privacy and confidentiality concerns in 6G Intelligent Networks is highlighted by the presence of diverse data owners and edge devices [25]. Federated Analytics (FA) emerges as a promising distributed computing paradigm to address these challenges, facilitating collaborative value generation from data while ensuring privacy and reducing communication overheads. FA offers significant advantages in managing and securing distributed and heterogeneous data networks within 6G systems. This paper examines FA principles and benefits, proposes an implementation framework tailored for 6G networks, and identifies research challenges and open issues.

Das et al. [26] have provided insights into the deployment of distributed learning over cellular networks, covering design aspects of FL for wireless communications, performance evaluation, and the impact of wireless factors on FL metrics. The article summarizes the advantages and obstacles associated with the utilization of FL schemes. Moreover, it analyzes promising FL-based technologies such as Mobile Edge Computing (MEC), satellite communications, semantic communications, Terahertz (THz) communications, network slicing, and blockchain for 6G-IoT networks. Furthermore, the fundamentals of decentralized Multi-Agent Reinforcement Learning (MARL)-based FL algorithms and operation principles are presented. The article integrates modern concepts into the MARL-enhanced FL framework for wireless networks, including power control mechanisms, interference mitigation, communication mode selection mechanisms, and resource management for handling large state and action spaces in dynamic wireless environments. Lastly, the article discusses potential barriers and outlines further research directions for applying MARL-based FL frameworks in 6G-IoT networks.

In this survey paper, our principal contributions are delineated as follows:

- We concentrate on solutions that concurrently bolster security and privacy within FL.
- A comprehensive comparison of these solutions is conducted using 13 distinct metrics.
- Our focus narrows to the specialized domain of mobile communication networks, where we categorize the proposed FL solutions tailored for this modern era.

- We conclude by pinpointing and deliberating on potential future research directions in this burgeoning field.

## 3 MOBILE COMMUNICATION AND FEDERATED LEARNING APPLICATION

At a high level, it would be stated that the mobile communication system consists of user equipment (UE), a radio access network (RAN), the core network (CN), a management layer (OAM), exposure layer, roaming interfaces, and business/operation support systems (OSS/BSS). All of these components except the OSS/BSS component are specified by the 3rd Generation Partnership Project (3GPP). Due to the vast and intricate architecture of the system, the implementation of AI/ML for addressing complex optimization challenges facilitates a decrease in operational costs. It ensures the provision of promising services with reduced emissions and the optimized allocation of resources, contributing to a sustainable world and enhancing the user experience.

The extensive integration of AI/ML in mobile communications commenced with the advent of 5G, encompassing not only proprietary solutions but also standardized ones. For instance, the 3GPP has formalized the application of AI/ML within core network management, as delineated in 3GPP TS 23.228, and within the Operation, Administration, and Maintenance (OAM) layer, as specified in 3GPP TS 28.105. In addition to the deployment of AI/ML, the 3GPP has further delineated the use of a collaborative ML approach, known as FL, in Release 18. Also, 3GPP started to study on incorporation of additional AI/ML mechanisms in Release 19, such as vertical FL.

The standardization of 5G systems is nearing completion, and with the advent of Release 20, the groundwork for 6G standardization studies will commence. The foundational elements for the 6G architecture have been identified in Hexa-X European Union research project, and the integration of these enablers is currently being explored in the design of a 6G architecture, as outlined in Hexa-X-II (the follow-up European Union research project of Hexa-X). The fulfillment of 6G use cases—encompassing a connected sustainable world, intelligent autonomous machines, the convergence of physical and cyber worlds, and the Internet of Senses—will necessitate a greater reliance on AI/ML solutions [27]–[29].

### 3.1 Federated Learning Solutions

In Section 2.2, we have mentioned existing surveys about the utilization of FL in mobile communication. In this subsection, we focus on the existing proposed solutions for usage of FL in mobile communication systems.

In 2020, Liu et al. proposed a framework for secure FL for 5G networks [30]. They focused on the two attacks: poisoning and membership inference attacks. They proposed a blockchain-based solution that prevents malicious actors from joining the FL setup and performing poisoning attacks. To mitigate the privacy concern, they propose local differential privacy. To increase the privacy level of FL usage in the core network, Zhou and Ansari propose the usage of partially homomorphic encryption in the NWDAF (Network Data Analytics Function) FL architecture [31]. They introduce a new entity for the generation and distribution of the keys. Phyu et al. 's study focuses on the RAN side for the use case of traffic forecasting [32]. They address the privacy concerns in the utilization of FL in the multi-slice setting. Hewa et. al. propose to use blockchain to make FL robust in 5G and beyond networks in their study [33]. Khowaja et al. present a comprehensive framework designed to assess the susceptibility of FL to poisoning and inversion attacks within 6G vehicular networks [34]. This analysis underscores the critical need for incorporating robust security and privacy measures in the deployment of FL technologies.

In 2023, Sanon, Reddy, et al. investigate computation on encrypted data technologies via secure multi-party computation for analysis of the network traffic data coming from different companies [35]. In addition to RAN, the core network, and specific vertical domains like vehicular networks, there is burgeoning research within the Open-RAN sector. Notably, one study concentrates on leveraging FL within the Open-RAN architecture to enhance automation capabilities [21].

Wasilewska et al. study the security of FL in the cognitive radio sensing use case [36]. Privacy in FL may especially become important if it is executed among multiple operators. Lan et al.'s study considers this multi-operator setup for the FL execution for performance prediction [37]. Another work that proposes to use block-chain for FL is the study of Moulahi et al. [38], which considers the threat intelligence scenario for cyber-threat detection. The study of Korba et al. [39] also uses FL for attack detection especially to detect zero-day attacks in 5G and beyond V2X networks. The study of Rubina Akter and Kim [40] focuses on the UAV-based beyond 5G networks and considers blockchain for FL.

Sharma et al. consider the localization for mass-beamforming beyond the 5G use case for the application of privacy-enhanced FL [41]. Rajabzadeh and Outtagarts focus on the core network NWDAF architecture [42]. Li et al. propose a framework based on blockchain to increase the trustworthiness level of FL in the Internet of Vehicles use case [43]. As mentioned before, one of the main benefits of AI/ML in mobile communication is the automation aspect. Saad et al. investigate how to secure the FL against poisoning attacks in the context of zero-touch beyond 5G communication systems [44].

The study of Zhang et al. takes the cell-free massive-MIMO use case and works on privacy aspects in the usage of FL in that use case [45]. In a recent study by Ayepah-Mensah et al., resource allocation and trading for network slicing is taken as a use case and blockchain usage for FL is investigated [46]. The study by Sanon, Lipps, et al. analyzes the effect of precision loss due to the usage of homomorphic encryption in a 5G wireless network traffic prediction use case [47]. The utilization of federated split learning was studied by Jiang et al. for the satellite-terrestrial integrated networks [48]. In most of the studies, FL is considered in the centralized setting where the setup consists of a central server and FL clients, but there are other approaches that do not need a centralized server. The study covers that aspect and proposes to use blockchaing [49]. I. A. Bartsiokas et al.'s study takes the multicellular next-generation network topologies into account and proposes to use FL for resource allocation use case [50].

RIS-based communication for collaborative computing in a swarm of drones setup is considered by Rahbari et al. [51] and they proposed the usage of FL for computation offloading. Danish Javeed, n.d. investigates Quantum-Empowered FL for 6G Wireless Networks for IoT Security and discusses the concepts, challenges, and future directions[52].

The study by Taghia et al. works on Congruent Learning in Self-regulated FL for the 6G system [53]. Reliability aspects of FL in the mmWave networks were investigated in [54], [55].

## 4 EXISTING GENERIC SOLUTIONS FOR SECURITY AND PRIVACY IN FEDERATED LEARNING

Before presenting details about existing solutions that resolve the security and privacy requirements simultaneously, we first present the metrics that we use in the comparison of these solutions in Table 2 and then compare the existing solutions in Table 3.

ELSA [57] proposes a solution for both security and privacy by utilizing secure multi-party computation and two non-colluding servers. To detect anomalies in the local model updates it uses the Norm Bounding approach. The solution works in the malicious adversary model but requires that at least one of the servers should be semi-honest to meet the assumption that the servers should not collude, i.e., unless both of them are infected and compromised, the privacy of peers is guaranteed. ELSA paper compares ELSA with six other custom (state-of-the-art FL) solutions and it is claimed that considering the comparison ELSA is accepted as superior to all of them. The paper also shares the cloud infrastructure details for the performance experiments, source codes, programming language, and libraries which enables the researchers to experiment with the scenarios. Although it addresses security and privacy, it doesn't meet the fairness expectations. The solution only guarantees malicious privacy and leaves the exploration of malicious security (privacy with correctness) for future work. Efficiently achieving malicious security seems quite challenging given that standard techniques aren't compelling for a large number of parties in the system.

Co-utility presented in [58] utilizes multi-hop communication topology and reputation-based approach to address security (e.g., such as Byzantine and Poisoning attacks) and privacy aspects simultaneously. They also claim that their solution is performance-efficient. The incentiveness approach is used to separate good and malicious participants. They compare their solution with the Homomorphic Encryption and Differential Privacy based ones. Despite presented tables and graphs that depict the solution's results, no source codes or test infrastructure info are shared.

Rofl [59] works in the single server model which is a more realistic scenario in most of the use cases, but they have to use zero-knowledge proofs to allow the server to validate that the clients' inputs, without violating privacy. Because of the usage of ZK proofs, the performance results of the solution are not good compared to the two-server-based solutions.

The Byzantine-Resilient Secure FL on Low-Bandwidth Networks proposal introduced in [60] also focuses on both security and privacy. Similar to ELSA they utilize secret sharing and distance nom bounds but the difference in that paper is that they don't need two or more servers but the clients secretly share their local model updates with each other and the security attack detection is performed on these shares.

Ensuring Integrity For Federated Learning (EIFFeL) in [61] is another recent work on security and privacy in FL. They formalize the problem of having privacy and integrity simultaneously in FL and propose a new solution. That solution validates the updates and removes them from the aggregation result, without allowing the server access to the updates. Similar to the Rofl, they don't require two non-colluding serves but need to utilize zero-knowledge proofs which reduces the performance of the solution.

SAFEFL introduced in [62] also focuses on privacy and poisoning attacks. They utilize secure multi-party computation to be able to address both the security and privacy aspects simultaneously.

DP-BREM addresses the privacy challenges by using differential privacy and secure aggregation and makes the FL process robust against Byzantine poisoning attacks by introducing the concept of client momentum which means taking averages of model updates of different rounds [63].

Flamingo [64] is based on a single-server solution that uses secure aggregation for privacy aspects. The clients encrypt the inputs by a masking operation and a small group of clients decrypts and communicates with the server. It also supports client drop-outs.

zPROBE [65] prefers to use high break point rank-based

**Table 2** Metrics for comparison of security&privacy solutions

| Metric | Acronym | Definition |
|---|---|---|
| Single server | SS | The solution does not require more than one server in the protocol |
| Multiple server | MS | The solution requires more than one server where at least one of the servers is semi-honest (i.e., all the servers cannot collude) |
| Semi-honest server | SHS | The solution is secure against the server that follows the protocol steps for the computation of the aggregation result |
| Malicious server | MCS | The solution is secure against the server that may not follow the protocol steps for the computation of the aggregation result |
| Aggregation integrity | AI | The solution ensures that the aggregation result is not altered by the server after the computation of the aggregation result. |
| Input privacy | IP | The solution does not leak information about the input of clients to the server(s) |
| Client drop-outs | CDO | The solution is robust against the client drop-outs (i.e., the secure aggregation can be computed even if some of the clients drops-out) |
| Input integrity | II | The solution ensures that the inputs of the clients are in a pre-defined input range |
| Semi-honest clients | SHC | The solution is secure against the client who follows the protocol steps after starting the protocol by providing their inputs |
| Malicious clients | MCC | The solution is secure against the clients who may not follow the protocol steps after starting the protocol by providing their inputs |
| Star topology | ST | The clients only need to communicate with the server |
| P2P topology | P2PT | The clients need to communicate with each other in addition to the server |
| Scalable | S | The solution computation and communication are linear both in the number of clients |

**Table 3** Comparison of security&privacy solutions

| Solution | SS | MS | SHS | MCS | AI | IP | CDO | II | SHC | MCC | ST | P2PT | S |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ELSA |  | ✓ | ✓ |  | ✓ | ✓ | ✓ | ✓ |  | ✓ |  | ✓ | ✓ |
| Co-utility | ✓ |  | ✓ |  |  | ✓ | ✓ | ✓ |  | ✓ |  | ✓ | ✓ |
| Rofl | ✓ |  | ✓ |  |  | ✓ | ✓ | ✓ |  | ✓ | ✓ |  | ✓ |
| Byzantine-Resilient | ✓ |  | ✓ |  |  | ✓ |  | ✓ | ✓ |  |  | ✓ | ✓ |
| EIFFeL | ✓ |  |  | ✓ | ✓ | ✓ | ✓ | ✓ |  | ✓ | ✓ |  | ✓ |
| SAFEFL |  | ✓ |  | ✓ | ✓ | ✓ | ✓ | ✓ |  | ✓ | ✓ |  | ✓ |
| DP-BREM | ✓ |  | ✓ |  |  | ✓ |  | ✓ | ✓ |  |  | ✓ | ✓ |
| Flamingo | ✓ |  |  | ✓ |  | ✓ | ✓ | ✓ |  | ✓ | ✓ |  | ✓ |
| zPROBE | ✓ |  | ✓ |  |  | ✓ |  | ✓ |  | ✓ | ✓ |  | ✓ |
| Prio |  | ✓ | ✓ |  |  | ✓ |  | ✓ |  | ✓ | ✓ |  | ✓ |
| Karakoç et al. [56] | ✓ |  | ✓ |  |  | ✓ | ✓ | ✓ |  | ✓ |  | ✓ | ✓ |

statistics on aggregated model updates instead of medians. They also utilize zero-knowledge protocols. With these approaches, they propose a solution that is secure against Byzantine type of attacks while still preserving privacy.

Prio [66] is also another that requires more than one server and has the assumption that at least one of the servers is semi-honest. They use secret sharing and secret-shared non-interactive proofs in their solution.

Another multi-hop communication-based solution was presented by Karakoç et al. [56]. To prevent malicious activities of the clients because of this anonymity, they propose to use partially blind signatures.

## 5 DISCUSSIONS AND CONCLUSION

While there has been progress in integrating ML into practical applications, there is still work to be done in order to improve security and privacy in these applications. Simultaneously addressing security and privacy in ML creates new potential for innovation as well as obstacles. Existing solutions frequently have drawbacks, such as the requirement for expensive and intricate cryptographic processes, dependence on numerous non-colluding servers, or dependence on peer-to-peer network topologies for communication. There are particular disadvantages associated with each of these strategies that may prevent their broad use and efficacy. Domain-specific research has investigated the use and application of ML with improvements to security and privacy; on the other hand, there is a significant lack of information in the literature about FL systems that simultaneously handle these issues. The distinct needs of different industries, like mobile communications, call for customized solutions that guarantee privacy and security. One area of research that shows promise is the use of FL in the telco industry. Examining use cases that require privacy and security at the same time may provide insightful information and innovative solutions. Depending on the particular use case, security and privacy needs can differ greatly, emphasising the necessity for adaptable and flexible solutions. Research ought to take into account the application of safe and privacy-enhanced ML in other sectors, in addition to telecommunications. Future research could examine how these specifications vary in various contexts and how FL can be modified to satisfy these changing requirements. All things considered, the way forward entails not only tackling the technical obstacles related to secure and privacy-enhanced ML but also comprehending the wider ramifications for many industries. In order to provide reliable solutions that can be successfully applied in practical applications, cooperation between academic institutions and businesses will be essential. The goal of future research should be to close the gaps that now exist and provide thorough frameworks that meet the various needs of various disciplines.

In conclusion, even though ML has made great progress in being incorporated into real-world applications, there is still more work to be done to create completely safe and private ML systems. It is particularly exciting to investigate FL as a way to accomplish these objectives simultaneously, especially in niche industries like telecoms. Future studies must tackle the distinct problems that different use cases provide in order to guarantee that the solutions are practical and flexible. We can open new possibilities and protect the integrity and privacy of data in ever more technical ways by pushing the limits of ML.

## ACKNOWLEDGEMENTS

This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) through the 1515 Frontier Research and Development Laboratories Support Program under Project 5169902.

## REFERENCES

[1] E. U. Soykan, L. Karaçay, F. Karakoç, and E. Tomur, “A survey and guideline on privacy enhancing technologies for collaborative machine learning,” *IEEE Access*, vol. 10, pp. 97 495–97 519, 2022. DOI: 10.1109/ACCESS.2022.3204037. [Online]. Available: https://doi.org/10.1109/ACCESS.2022.3204037.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282.

[3] P. Kairouz, H. B. McMahan, B. Avent, *et al.*, “Advances and open problems in federated learning,” *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[4] D. Cao, S. Chang, Z. Lin, G. Liu, and D. Sun, “Understanding distributed poisoning attack in federated learning,” in *2019 IEEE 25th international conference on parallel and distributed systems (ICPADS)*, IEEE, 2019, pp. 233–239.

[5] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, “A survey on security and privacy of federated learning,” *Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, 2021. DOI: 10.1016/J.FUTURE.2020.10.007. [Online]. Available: https://doi.org/10.1016/j.future.2020.10.007.

[6] A. Blanco-Justicia, J. Domingo-Ferrer, S. Martínez, D. Sánchez, A. Flanagan, and K. E. Tan, “Achieving security and privacy in federated learning systems: Survey, research challenges and future directions,” *Eng. Appl. Artif. Intell.*, vol. 106, p. 104 468, 2021. DOI: 10.1016/J.ENGAPPAI.2021.104468. [Online].

Available: https://doi.org/10.1016/j.engappai.2021.104468.

[7] N. B. Truong, K. Sun, S. Wang, F. Guitton, and Y. Guo, "Privacy preservation in federated learning: An insightful survey from the GDPR perspective," *Comput. Secur.*, vol. 110, p. 102 402, 2021. DOI: 10.1016/J.COSE.2021.102402. [Online]. Available: https://doi.org/10.1016/j.cose.2021.102402.

[8] N. Bouacida and P. Mohapatra, "Vulnerabilities in federated learning," *IEEE Access*, vol. 9, pp. 63 229–63 249, 2021. DOI: 10.1109/ACCESS.2021.3075203. [Online]. Available: https://doi.org/10.1109/ACCESS.2021.3075203.

[9] M. Mansouri, M. Önen, W. B. Jaballah, and M. Conti, "SoK: Secure aggregation based on cryptographic schemes for federated learning," *Proc. Priv. Enhancing Technol.*, vol. 2023, no. 1, pp. 140–157, 2023. DOI: 10.56553/POPETS-2023-0009. [Online]. Available: https://doi.org/10.56553/popets-2023-0009.

[10] D. Enthoven and Z. Al-Ars, "An overview of federated deep learning privacy attacks and defensive strategies," *CoRR*, vol. abs/2004.04676, 2020. arXiv: 2004.04676. [Online]. Available: https://arxiv.org/abs/2004.04676.

[11] L. Lyu, H. Yu, X. Ma, *et al.*, "Privacy and robustness in federated learning: Attacks and defenses," *CoRR*, vol. abs/2012.06337, 2020. arXiv: 2012.06337. [Online]. Available: https://arxiv.org/abs/2012.06337.

[12] J. Mao, C. Cao, L. Wang, J. Ye, and W. Zhong, "Research on the security technology of federated learning privacy preserving," *Journal of Physics: Conference Series*, vol. 1757, no. 1, p. 012 192, Jan. 2021. DOI: 10.1088/1742-6596/1757/1/012192. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/1757/1/012192.

[13] M. Asad, S. Shaukat, D. Hu, *et al.*, "Limitations and future aspects of communication costs in federated learning: A survey," *Sensors*, vol. 23, no. 17, p. 7358, 2023. DOI: 10.3390/S23177358. [Online]. Available: https://doi.org/10.3390/s23177358.

[14] A. Akhtarshenas, M. A. Vahedifar, N. Ayoobi, B. Maham, T. Alizadeh, and S. Ebrahimi, "Federated learning: A cutting-edge survey of the latest advancements and applications," *CoRR*, vol. abs/2310.05269, 2023. DOI: 10.48550/ARXIV.2310.05269. arXiv: 2310.05269. [Online]. Available: https://doi.org/10.48550/arXiv.2310.05269.

[15] D. Sirohi, N. Kumar, P. S. Rana, S. Tanwar, R. Iqbal, and M. Hijji, "Federated learning for 6G-enabled secure communication systems: A comprehensive survey," *Artif. Intell. Rev.*, vol. 56, no. 10, pp. 11 297–11 389, 2023. DOI: 10.1007/S10462-023-10417-3. [Online]. Available: https://doi.org/10.1007/s10462-023-10417-3.

[16] M. Al-Quraan, L. S. Mohjazi, L. Bariah, *et al.*, "Edge-native intelligence for 6G communications driven by federated learning: A survey of trends and challenges," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 7, no. 3, pp. 957–979, 2023. DOI: 10.1109/TETCI.2023.3251404. [Online]. Available: https://doi.org/10.1109/TETCI.2023.3251404.

[17] Y. Liu, X. Yuan, Z. Xiong, J. Kang, X. Wang, and D. Niyato, "Federated learning for 6G communications: Challenges, methods, and future directions," *CoRR*, vol. abs/2006.02931, 2020. arXiv: 2006.02931. [Online]. Available: https://arxiv.org/abs/2006.02931.

[18] A. Rahman, K. Hasan, D. Kundu, *et al.*, "On the ICN-IoT with federated learning integration of communication: Concepts, security-privacy issues, applications, and future perspectives," *Future Gener. Comput. Syst.*, vol. 138, pp. 61–88, 2023. DOI: 10.1016/J.FUTURE.2022.08.004. [Online]. Available: https://doi.org/10.1016/j.future.2022.08.004.

[19] Y. Zuo, J. Guo, N. Gao, Y. Zhu, S. Jin, and X. Li, "A survey of blockchain and artificial intelligence for 6G wireless communications," *IEEE Commun. Surv. Tutorials*, vol. 25, no. 4, pp. 2494–2528, 2023. DOI: 10.1109/COMST.2023.3315374. [Online]. Available: https://doi.org/10.1109/COMST.2023.3315374.

[20] S. Abimannan, E.-S. M. El-Alfy, S. Hussain, *et al.*, "Towards federated learning and multi-access edge computing for air quality monitoring: Literature review and assessment," *Sustainability*, vol. 15, no. 18, 2023, ISSN: 2071-1050. DOI: 10.3390/su151813951. [Online]. Available: https://www.mdpi.com/2071-1050/15/18/13951.

[21] N. A. Khalek, D. H. Tashman, and W. Hamouda, "Advances in machine learning-driven cognitive radio for wireless networks: A survey," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2023. DOI: 10.1109/COMST.2023.3345796.

[22] M. B. Driss, E. Sabir, H. Elbiaze, and W. Saad, "Federated learning for 6G: Paradigms, taxonomy, recent advances and insights," *CoRR*, vol. abs/2312.04688, 2023. DOI: 10.48550/ARXIV.2312.04688. arXiv: 2312.04688. [Online]. Available: https://doi.org/10.48550/arXiv.2312.04688.

[23] M. A. Ferrag, O. Friha, B. Kantarci, *et al.*, “Edge learning for 6G-enabled internet of things: A comprehensive survey of vulnerabilities, datasets, and defenses,” *IEEE Commun. Surv. Tutorials*, vol. 25, no. 4, pp. 2654–2713, 2023. DOI: 10.1109/COMST.2023.3317242. [Online]. Available: https://doi.org/10.1109/COMST.2023.3317242.

[24] I. Bartsiokas, P. Gkonis, A. Papazafeiropoulos, D. Kaklamani, and I. Venieris, “Federated learning for 6G hetnets’ physical layer optimization: Perspectives, trends, and challenges federated learning for 6G hetnets’ physical layer optimization,” in Jul. 2024, p. 1–28, ISBN: 9781668473665. DOI: 10.4018/978-1-6684-7366-5.ch070.

[25] J. M. P. Ullauri, X. Zhang, A. Bravalheri, Y. Wu, R. Nejabati, and D. Simeonidou, “Federated analytics for 6G networks: Applications, challenges, and opportunities,” *CoRR*, vol. abs/2401.03878, 2024. DOI: 10.48550/ARXIV.2401.03878. arXiv: 2401.03878. [Online]. Available: https://doi.org/10.48550/arXiv.2401.03878.

[26] S. K. Das, R. Mudi, M. S. Rahman, and A. O. Fapojuwo, “Distributed learning for 6G–IoT networks: A comprehensive survey,” *Authorea Preprints*, 2023.

[27] L. S. Mohjazi, B. Selim, M. Tatipamula, and M. A. Imran, “The journey towards 6G: A digital and societal revolution in the making,” *CoRR*, vol. abs/2306.00832, 2023. DOI: 10.48550/ARXIV.2306.00832. arXiv: 2306.00832. [Online]. Available: https://doi.org/10.48550/arXiv.2306.00832.

[28] C. Anitha, B. Balakiruthiga, S. Angayarkanni, P. P. Selvi, and L. S. Kumar, “Recent developments, application cases, and lingering issues on the path to a 6G IoT,” in *2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering (RMKMATE)*, IEEE, 2023, pp. 1–10.

[29] S. Polymeni, S. Plastras, D. N. Skoutas, G. Kormentzas, and C. Skianis, “The impact of 6G-IoT technologies on the development of agriculture 5.0: A review,” *Electronics*, vol. 12, no. 12, 2023, ISSN: 2079-9292. DOI: 10.3390/electronics12122651. [Online]. Available: https://www.mdpi.com/2079-9292/12/12/2651.

[30] Y. Liu, J. Peng, J. Kang, A. M. Iliyasu, D. Niyato, and A. A. A. El-Latif, “A secure federated learning framework for 5G networks,” *CoRR*, vol. abs/2005.05752, 2020. arXiv: 2005.05752. [Online]. Available: https://arxiv.org/abs/2005.05752.

[31] C. Zhou and N. Ansari, “Securing federated learning enabled NWDAF architecture with partial homomorphic encryption,” *IEEE Netw. Lett.*, vol. 5, no. 4, pp. 299–303, 2023. DOI: 10.1109/LNET.2023.3294497. [Online]. Available: https://doi.org/10.1109/LNET.2023.3294497.

[32] H. P. Phyu, R. Stanica, and D. Naboulsi, “Multi-slice privacy-aware traffic forecasting at RAN level: A scalable federated-learning approach,” *IEEE Trans. Netw. Serv. Manag.*, vol. 20, no. 4, pp. 5038–5052, 2023. DOI: 10.1109/TNSM.2023.3267725. [Online]. Available: https://doi.org/10.1109/TNSM.2023.3267725.

[33] T. Hewa, P. Porambage, M. Liyanage, and M. Ylianttila, “Towards attack resistant federated learning with blockchain in 5G and beyond networks,” in *2023 Joint European Conference on Networks and Communications & 6G Summit, EuCNC/6G Summit 2023, Gothenburg, Sweden, June 6-9, 2023*, Jun. 2023.

[34] S. A. Khowaja, P. Khuwaja, K. Dev, and A. Antonopoulos, “SPIN: Simulated poisoning and inversion network for federated learning-based 6G vehicular networks,” in *IEEE International Conference on Communications, ICC 2023, Rome, Italy, May 28 - June 1, 2023*, IEEE, 2023, pp. 6205–6210. DOI: 10.1109/ICC45041.2023.10279339. [Online]. Available: https://doi.org/10.1109/ICC45041.2023.10279339.

[35] S. P. Sanon, R. Reddy, C. Lipps, and H. D. Schotten, “Secure federated learning: An evaluation of homomorphic encrypted network traffic prediction,” in *20th IEEE Consumer Communications & Networking Conference, CCNC 2023, Las Vegas, NV, USA, January 8-11, 2023*, IEEE, 2023, pp. 1–6. DOI: 10.1109/CCNC51644.2023.10060116. [Online]. Available: https://doi.org/10.1109/CCNC51644.2023.10060116.

[36] M. Wasilewska, H. Bogucka, and H. V. Poor, “Secure federated learning for cognitive radio sensing,” *CoRR*, vol. abs/2304.06519, 2023. DOI: 10.48550/ARXIV.2304.06519. arXiv: 2304.06519. [Online]. Available: https://doi.org/10.48550/arXiv.2304.06519.

[37] X. Lan, J. Taghia, F. Moradi, *et al.*, “Federated learning for performance prediction in multi-operator environments,” *ITU Journal on Future and Evolving Technologies*, vol. 4, pp. 166–177, Mar. 2023. DOI: 10.52953/PFYZ9165.

[38] T. Moulahi, R. Jabbar, A. Alabdulatif, *et al.*, “Privacy-preserving federated learning cyber-threat detection for intelligent transport systems with blockchain-based security,” *Expert Syst. J. Knowl. Eng.*, vol. 40,

no. 5, 2023. DOI: 10.1111/EXSY.13103. [Online]. Available: https://doi.org/10.1111/exsy.13103.

[39] A. A. Korba, A. Boualouache, B. Brik, R. Rahal, Y. Ghamri-Doudane, and S. M. Senouci, "Federated learning for zero-day attack detection in 5G and beyond V2X networks," in *IEEE International Conference on Communications, ICC 2023, Rome, Italy, May 28 - June 1, 2023*, IEEE, 2023, pp. 1137–1142. DOI: 10.1109/ICC45041.2023.10279368. [Online]. Available: https://doi.org/10.1109/ICC45041.2023.10279368.

[40] A. Z. Rubina Akter and D.-S. Kim, "UAV-based B5G networks: Blockchain and federated learning technology," 2023.

[41] D. Sharma, A. Kumar, and R. B. Battula, "Fedbeam: Federated learning based privacy preserved localization for mass-beamforming in 5GB," in *International Conference on Information Networking, ICOIN 2023, Bangkok, Thailand, January 11-14, 2023*, IEEE, 2023, pp. 616–621. DOI: 10.1109/ICOIN56518.2023.10048980. [Online]. Available: https://doi.org/10.1109/ICOIN56518.2023.10048980.

[42] P. Rajabzadeh and A. Outtagarts, "Federated learning for distributed NWDAF architecture," in *26th Conference on Innovation in Clouds, Internet and Networks, ICIN 2023, Paris, France, March 6-9, 2023*, pp. 24–26. DOI: 10.1109/ICIN56760.2023.10073493. [Online]. Available: https://doi.org/10.1109/ICIN56760.2023.10073493.

[43] A. Li, X. Chang, J. Ma, S. Sun, and Y. Yu, "VTFL: A blockchain based vehicular trustworthy federated learning framework," in *2023 IEEE 6th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*, vol. 6, 2023, pp. 1002–1006. DOI: 10.1109/ITNEC56291.2023.10082698.

[44] S. B. Saad, B. Brik, and A. Ksentini, "Toward securing federated learning against poisoning attacks in zero touch B5G networks," *IEEE Trans. Netw. Serv. Manag.*, vol. 20, no. 2, pp. 1612–1624, 2023. DOI: 10.1109/TNSM.2023.3278838. [Online]. Available: https://doi.org/10.1109/TNSM.2023.3278838.

[45] J. Zhang, J. Zhang, D. W. K. Ng, and B. Ai, "Federated learning-based cell-free massive MIMO system for privacy-preserving," *IEEE Trans. Wirel. Commun.*, vol. 22, no. 7, pp. 4449–4460, 2023. DOI: 10.1109/TWC.2022.3225812. [Online]. Available: https://doi.org/10.1109/TWC.2022.3225812.

[46] D. Ayepah-Mensah, G. Sun, G. O. Boateng, S. Anokye, and G. Liu, "Blockchain-enabled federated learning-based resource allocation and trading for network slicing in 5G," *IEEE/ACM Trans. Netw.*, vol. 32, no. 1, pp. 654–669, 2024. DOI: 10.1109/TNET.2023.3297390. [Online]. Available: https://doi.org/10.1109/TNET.2023.3297390.

[47] S. P. Sanon, C. Lipps, and H. D. Schotten, "Fully homomorphic encryption: Precision loss in wireless mobile communication," in *2023 Joint European Conference on Networks and Communications & 6G Summit, EuCNC/6G Summit 2023, Gothenburg, Sweden, June 6-9, 2023*, IEEE, 2023, pp. 466–471. DOI: 10.1109/EUCNC/6GSUMMIT58263.2023.10188286. [Online]. Available: https://doi.org/10.1109/EuCNC/6GSummit58263.2023.10188286.

[48] W. Jiang, H. Han, Y. Zhang, and J. Mu, "Federated split learning for sequential data in satellite-terrestrial integrated networks," *Inf. Fusion*, vol. 103, p. 102 141, 2024. DOI: 10.1016/J.INFFUS.2023.102141. [Online]. Available: https://doi.org/10.1016/j.inffus.2023.102141.

[49] F. Wilhelmi, L. Giupponi, and P. Dini, "Blockchain-enabled Server-less Federated Learning," *CoRR*, vol. abs/2112.07938, 2021. arXiv: 2112.07938. [Online]. Available: https://arxiv.org/abs/2112.07938.

[50] I. A. Bartsiokas, P. K. Gkonis, D. I. Kaklamani, and I. S. Venieris, "A federated learning-based resource allocation scheme for relaying-assisted communications in multicellular next generation network topologies," *Electronics*, vol. 13, no. 2, 2024, ISSN: 2079-9292. DOI: 10.3390/electronics13020390. [Online]. Available: https://www.mdpi.com/2079-9292/13/2/390.

[51] D. Rahbari, M. M. Alam, Y. L. Moullec, and M. Jenihhin, "Applying RIS-based communication for collaborative computing in a swarm of drones," *IEEE Access*, vol. 11, pp. 70 093–70 109, 2023. DOI: 10.1109/ACCESS.2023.3293737. [Online]. Available: https://doi.org/10.1109/ACCESS.2023.3293737.

[52] D. Javeed, M. Saeed, I. Ahmad, M. Adil, P. Kumar, and N. Islam, "Quantum-empowered federated learning and 6G wireless networks for IoT security: Concept, challenges and future directions," *Future Generation Computer Systems*, Jun. 2024. DOI: 10.1016/j.future.2024.06.023.

[53] J. Taghia, F. Moradi, H. Larsson, *et al.*, "Congruent learning for self-regulated federated learning in 6G," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 2, pp. 129–149, 2024. DOI: 10.1109/TMLCN.2023.3347680.

[54] M. Al-Quraan, A. Zoha, A. Centeno, *et al.*, "Enhancing reliability in federated mmwave networks: A practical and scalable solution using radar-aided dynamic blockage recognition," *CoRR*, vol. abs/2307.06834, 2023. DOI: 10.48550/ARXIV.2307.06834. arXiv: 2307.06834. [Online]. Available: https://doi.org/10.48550/arXiv.2307.06834.

[55] M. Al-Quraan, A. Centeno, A. Zoha, M. A. Imran, and L. S. Mohjazi, "Federated learning for reliable mmwave systems: Vision-aided dynamic blockages prediction," in *IEEE Wireless Communications and Networking Conference, WCNC 2023, Glasgow, UK, March 26-29, 2023*, IEEE, 2023, pp. 1–6. DOI: 10.1109/WCNC55385.2023.10118675. [Online]. Available: https://doi.org/10.1109/WCNC55385.2023.10118675.

[56] F. Karakoç, L. Karaçay, P. Ç. D. Cnudde, U. Gülen, R. Fuladi, and E. U. Soykan, "A security-friendly privacy-preserving solution for federated learning," *Comput. Commun.*, vol. 207, pp. 27–35, 2023. DOI: 10.1016/j.comcom.2023.05.004. [Online]. Available: https://doi.org/10.1016/j.comcom.2023.05.004.

[57] M. Rathee, C. Shen, S. Wagh, and R. A. Popa, "ELSA: Secure aggregation for federated learning with malicious actors," *IACR Cryptol. ePrint Arch.*, p. 1695, 2022. [Online]. Available: https://eprint.iacr.org/2022/1695.

[58] J. Domingo-Ferrer, A. Blanco-Justicia, J. A. Manjón, and D. Sánchez, "Secure and privacy-preserving federated learning via co-utility," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3988–4000, 2022. DOI: 10.1109/JIOT.2021.3102155. [Online]. Available: https://doi.org/10.1109/JIOT.2021.3102155.

[59] H. Lycklama, L. Burkhalter, A. Viand, N. Küchler, and A. Hithnawi, "Rofl: Robustness of secure federated learning," in *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*, IEEE, 2023, pp. 453–476. DOI: 10.1109/SP46215.2023.10179400. [Online]. Available: https://doi.org/10.1109/SP46215.2023.10179400.

[60] H. Masuda, K. Kita, Y. Koizumi, J. Takemasa, and T. Hasegawa, "Byzantine-resilient secure federated learning on low-bandwidth networks," *IEEE Access*, vol. 11, pp. 51 754–51 766, 2023. DOI: 10.1109/ACCESS.2023.3277858.

[61] A. R. Chowdhury, C. Guo, S. Jha, and L. van der Maaten, "EIFFeL: Ensuring integrity for federated learning," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, pp. 2535–2549. DOI: 10.1145/3548606.3560611. [Online]. Available: https://doi.org/10.1145/3548606.3560611.

[62] T. Gehlhar, F. Marx, T. Schneider, A. Suresh, T. Wehrle, and H. Yalame, "SafeFL: MPC-friendly framework for private and robust federated learning," in *2023 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, May 25, 2023*, IEEE, 2023, pp. 69–76. DOI: 10.1109/SPW59333.2023.00012. [Online]. Available: https://doi.org/10.1109/SPW59333.2023.00012.

[63] X. Gu, M. Li, and L. Xiong, "DP-BREM: Differentially-private and byzantine-robust federated learning with client momentum," *CoRR*, vol. abs/2306.12608, 2023. DOI: 10.48550/ARXIV.2306.12608. arXiv: 2306.12608. [Online]. Available: https://doi.org/10.48550/arXiv.2306.12608.

[64] Y. Ma, J. Woods, S. Angel, A. Polychroniadou, and T. Rabin, "Flamingo: Multi-round single-server secure aggregation with applications to private federated learning," in *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*, IEEE, 2023, pp. 477–496. DOI: 10.1109/SP46215.2023.10179434. [Online]. Available: https://doi.org/10.1109/SP46215.2023.10179434.

[65] Z. Ghodsi, M. Javaheripi, N. Sheybani, X. Zhang, K. Huang, and F. Koushanfar, "zPROBE: Zero peek robustness checks for federated learning," in *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, IEEE, 2023, pp. 4837–4847. DOI: 10.1109/ICCV51070.2023.00448. [Online]. Available: https://doi.org/10.1109/ICCV51070.2023.00448.

[66] H. Corrigan-Gibbs and D. Boneh, "Prio: Private, robust, and scalable computation of aggregate statistics," *CoRR*, vol. abs/1703.06255, 2017. arXiv: 1703.06255. [Online]. Available: http://arxiv.org/abs/1703.06255.

ITU/JWCC

# Real-Time Digital Twin Platform: A Case Study on Core Network Selection in Aeronautical Ad-hoc Networks

**Lal Verda Cakir**[1,2], **Mihriban Kocak**[3], **Mehmet Özdem**[4], **and Berk Canberk**[1,5]

[1] School of Computing, Engineering and The Built Environment, Edinburgh Napier University, EH10 5DT, United Kingdom
[2] BTS Group, Istanbul, 34469, Turkey
[3] Department of Computer Science, Aalto University, 02150 Espoo, Finland
[4] Turk Telekom, Istanbul, 344469, Turkey
[5] Department of Artificial Intelligence and Data Engineering, Istanbul Technical University, Istanbul, 34469, Turkey

**Abstract:** The development of Digital Twins (DTs) is hindered by a lack of specialized, open-source solutions that can meet the demands of dynamic applications. This has caused state-of-the-art DT applications to be validated using offline data. However, this approach falls short of integrating real-time data, which is one of the most important characteristics of DTs. This can limit the validating effectiveness of DT applications in cases such as aeronautical ad-hoc networks (AANETs). Considering this, we develop a Real-Time Digital Twin Platform and implement core network selection in AANETs as a case study. In this, we implement microservice-based architecture and design a robust data pipeline. Additionally, we develop an interactive user interface using open-source tools. Using this, the platform supports real-time decision-making in the presence of data retrieval failures.

**Keywords:** Digital twin, real-time platform, aeronautical networks, data pipeline.

## Gerçek Zamanlı Dijital İkiz Platformu: Ad-hoc Hava Ağlarında Çekirdek Ağ Seçimi Üzerine Vaka Çalışması

**Özet:** Dijital İkizlerin (Dİ) geliştirilmesi için dinamik uygulamaların taleplerini karşılayabilme kabiliyetindeki ve açık kaynaklı çözümlerin eksiklikliklerden dolayı sekteye uğramaktadır. Bu, son teknoloji Dİ uygulamalarının çevrimdışı veriler kullanılarak doğrulanmasına neden olmuştur. Ancak bu yaklaşım, Dİ'lerin en önemli özelliklerinden biri olan gerçek zamanlı verilerin entegrasyonu konusunda yetersizdir. Bu, ad-hoc hava ağları (AHHA) gibi vakalarda Dİ uygulamalarının doğrulanmasını sınırlayabilmektedir. Bunu göz önünde bulundurarak, bu çalışmada Gerçek Zamanlı Dijital İkiz Platformu geliştirilmiş ve örnek olay olarak AHHA'larda çekirdek ağ seçimini uygulanmıştır. Bunda mikroservis tabanlı mimariye sahip dayanıklı veri işleme hattı tasarlanmıştır. Ayrıca açık kaynaklı araçlar kullanarak etkileşimli bir kullanıcı arayüzü geliştirilmiştir. Bunlar sayesinde geliştirilen platform, veri alma hataları durumunda dahi gerçek zamanlı karar almayı destekler hale getirilebilmiştir.

**Anahtar Kelimeler:** Dijital ikiz, gerçek zamanlı platform, hava ağları, veri işleme hattı.

RESEARCH PAPER

Corresponding Author: Lal Verda Cakir, lal.cakir@napier.ac.uk

Reference: L. V. Cakir, M. Kocak, M. Özdem, B. Canberk, (2024), "Real-Time Digital Twin Platform: A Case Study on Core Network Selection in Aeronautical Ad-hoc Networks," *ITU-Journ. Wireless Comm. Cyber.*, 1, (1) 41–46.

Submission Date: Aug, 19, 2024
Acceptance Date: Sept, 26, 2024
Online Publishing: Sept, 30, 2024

## 1 INTRODUCTION

The DT field has limited availability of platforms that cater to the diverse and complex requirements of various applications. Existing platforms are often too generic, lacking the specific features, capabilities, and programmability needed to support highly dynamic and intricate environments [1]. Additionally, many of these are not open-sourced, restricting access for researchers and developers. This shortage can significantly impact the ability to perform a comparative evaluation of DTs [2]. Due to this, existing studies on DTs have frequently relied on pre-existing data in closed environments. This lack of real-time data integration limits the ability to accurately reflect the current state and ensure the testing is valid for real-world implementations. This challenge is especially prominent in applications that require immediate responses, such as core network selection in aeronautical ad-hoc networks (AANETs). To ensure the continuous connectivity of AANETs, the incoming data must be processed, and a new decision should be made at the right time without interrupting the service. While this can be realised to some degree using the existing data within the closed environment, a new set of challenges can arise when real-time data is integrated [3], [4]. For instance, the discrepancies and irregularities in the data collection may cause the DT model to be in temporal misalignment[5]. At this, applying estimation of these unknown factors has been a key approach [6]. Therefore, the solutions must be tested beforehand by means of integrating real-time data to identify, address, and overcome such obstacles.

### 1.1 Related Work

The DT technology has grown in popularity due to its ability to mimic, emulate, or replicate the characteristics or specific behaviour of corresponding physical entities. This technology corresponds to creating a software model that can create real-time virtual representations of an object, process or system [7], [8]. The capabilities of DTs are supported by the rapid development of Artificial Intelligence (AI) and Machine Learning (ML) technologies [9]. With this, DTs differentiate from classic simulation models by constantly learning and adapting [10]. In this way, predictions for the future can be made and directly integrated into management's decision-making process.

Moreover, DTs have become an important technology in the field of aviation connectivity. In this, aeronautical ad-hoc networks (AANETs) are widely used by commercial and military aviation organizations to facilitate communication. This connectivity is crucial to allow for data exchange of flight plans and weather updates between aircraft. Moreover, according to the survey results in [11] 66% of the passengers claim that in-flight connectivity (IFC) is an essential requirement while it was seen as a luxury service in the past. Overall, AANETs are required to fulfil the high-quality internet connection demands of aviation organizations. To address this, in [12], the DTs are used to increase the efficiency of core network selection in WIFI-enabled in-flight connectivity (W-IFC). This study has used the recorded data from [13], which is used in sampled periods offline. While the results acquired validated the potential of DTs in this field, the proposed architecture disregarded the streaming nature of the data. Here, developing the interaction mechanisms and related interfaces is highly important to ensure the DT model can operate accurately [14]. As also was pointed out in the reference DT platform in [15], the core functionalities of connection, DT interaction and DT visualizations are necessary. This use of microservice-based architecture offers a promising solution to ease the development and deployment of this platform [16]

### 1.2 Contributions

Considering these, we develop the Real-Time Digital Twin Platform with a case study on core network selection in AANETs and contribute to the literature as follows:

- We develop the real-time digital twin platform using open-source tools and leveraging microservice-based architecture. In this, we use a time-series database as the data store and develop a flask application for an interactive user interface.
- We implement the core network selection in AANETs as a case study and use the Openskynetwork API to retrieve real-time data on the aircraft. Thanks to testing with real-time data integration, we reveal that the API can fail to return data in some instances due to unknown factors. This is a highly possible case in real-world scenarios that can occur due to factors such as congestion, failures and backlogs.
- The robust data pipeline is developed using a stream processing engine, Kapacitor, to support real-time decision-making. In this, the feature scaling, clustering, and recommendation operations of core network selection are performed streamingly. We design this pipeline so that it can handle the case of an extract operation failing to return or empty data. For this, we implement the projection module, which is activated to calculate the information on the aircraft. This robust design can allow core network selections to continue without disruption to the service.

### 1.3 Organization of the Article

The remainder of this article is structured as follows: Section 2 presents the proposed Real-Time Digital Twin Platform, and Section 3 provides the performance evaluations. Then, we conclude the article and provide a discussion on future works in Section 4.

## 2 REAL-TIME DIGITAL TWIN PLATFORM

In this study, we design the Real-Time DT Platform using a three-layered approach, which includes Physical, DT, and Service Layers as illustrated in the Figure 1. In the design of this platform, we opt for microservice-based architecture over a monolithic one, considering its performance implications [17]. Moreover, monolithic architectures are often built with a single and unified codebase, which may force any change to one part of the system, often requiring redeploying the entire platform. For a real-time DT platform, this would reduce flexibility and increase the risk of failure. Also, improving in a monolithic style and managing and debugging the platform can become complex as the platform grows.

**Table 1** Data Retrieved from OpenSky Network API [18]

| Field Name | Description |
|---|---|
| $icao24$ | Transponder Address |
| $callsign$ | Callsign of Aircraft |
| $origin\_country$ | Country Name |
| $time\_position$ | Timestamp of Position |
| $last\_contact$ | Timestamp for Update |
| $longitude$ | WGS-84 Longitude |
| $latitude$ | WGS-84 Latitude |
| $baro\_altitude$ | Barometric Altitude (m) |
| $on\_ground$ | If surface position report |
| $velocity$ | Ground Velocity of aircraft (m/s) |
| $true\_track$ | Direction of Aircraft |
| $vertical\_rate$ | vertical velocity of Aircraft (m/s) |
| $geo\_altitude$ | Geometric Altitude |
| $spi$ | Flight Status |
| $category$ | Aircraft Category |

### 2.1 Physical Layer

The Physical Layer consists of the entities and processes in which their DT is created. In the scope of this article, we consider the core network selection in the AANETs use case and specifically chose the UK area because of its active and diverse air traffic, which carries the main characteristics of AANETS. Accordingly, we retrieve real-time air traffic data from this layer using the OpenSky Network [19]. The details of the data are given in Table 1.

### 2.2 Digital Twin (DT) Layer

At the DT Layer, the DTs are created and maintained using the real-time data coming from the Physical Layer and their data is stored in the time-series database, InfluxDB [20]. In our DT Platform, this process is handled by the Robust Data Pipeline design, which is responsible for extracting, transforming and loading (ETL) operations and handling the disruption in the data flow.

#### 2.2.1 Robust data pipeline

The real-time core network selection in our DT Implementation is performed according to the following data pipeline as illustrated in Figure 1.

1. **Extract**: The data that the details given in Table 1 is retrieved repetitively from the API. Here, in case the incoming data is empty, the last records on the aircraft are extracted from the time-series database.

2. **Transform**: The incoming data undergoes the following operation steps.

   (a) **Projection:** The timestamp of the position ($time\_position$) in the data may not be the same for each aircraft. To account for that, the location is corrected using the following equations:

   $$\Delta x = velocity * cos(true\_track) * \Delta t \tag{1}$$

   $$\Delta y = velocity * sin(true\_track) * \Delta t \tag{2}$$

   $$\Delta z = vertical\_rate * \Delta t \tag{3}$$

   (b) **Feature scaling:** We scale features to a range of 0 to 1 using the Min-Max Scaler from the Scikit-learn library [21].

   (c) **Bayesian Information Criterion (BIC):** Before clustering, we decide the optimal number of clusters (k) using the Bayesian Information Criterion (BIC) method, which is claimed as a superior alternative to the elbow method [22].

   (d) **Clustering:** We perform clustering using the K-means algorithm thanks to its simplicity and effectiveness in similarity-based clustering. We use the KMeans class from the Scikit-learn library [21] over the scaled data in the previous step.

   (e) **Recommendation:** We perform recommendations for the core network selection based on the methodology introduced in the [12].

3. **Load**: The results are loaded into the time-series database with the buckets Physical and DT. Here, the Physical bucket stores only the data retrieved from the

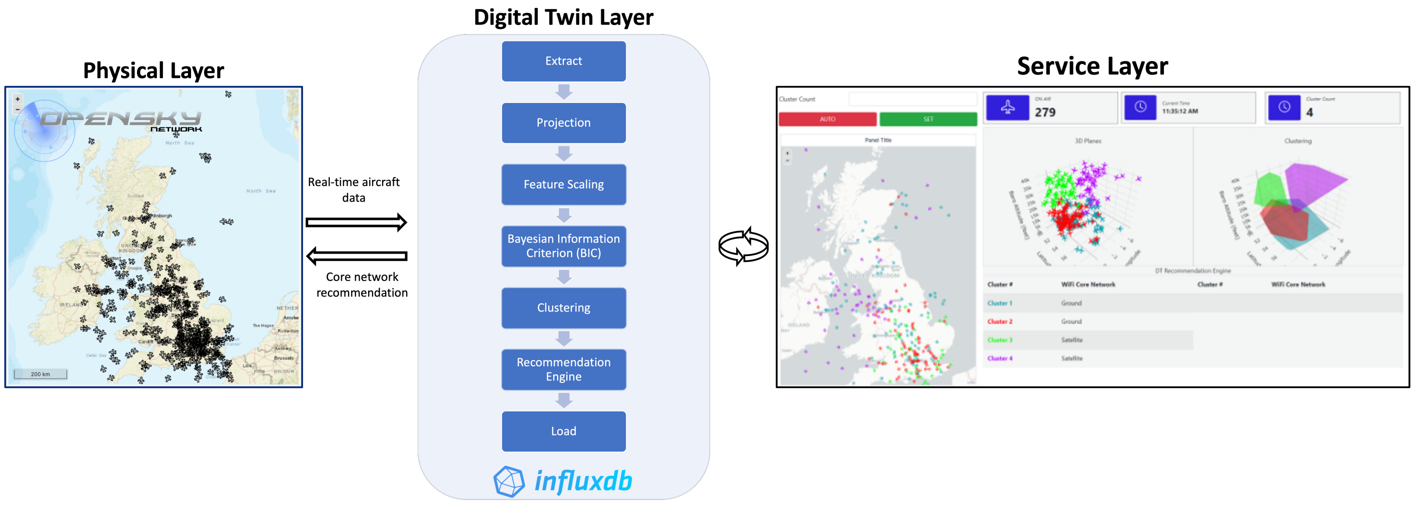


**Fig. 1** The three-layered architecture of real-time digital twin platform with robust data pipeline.

API, while the DT bucket stores the data after projection, cluster assignment and the corresponding recommendation. Here, the data is represented as line protocols that are in the following form:

$$measurement\ tag, tag1, \ldots\ field, field3, \ldots\ timestamp \quad (4)$$

## 2.3 Service Layer

In the Service Layer of our Real-Time Digital Twin Platform, we develop a web application interface using Flask framework [23] and Grafana [24] and Plotly [25]. By this, the monitoring, clustering results, and recommendations are visualised in an interactive dashboard as shown in Figure 1. Within this, aircraft clusters are displayed through three different dashboards in which the aircraft are colour-coded based on their respective clusters. The first dashboard portrays aircraft on a map with their 2D coordinates (longitude and latitude) determining their positions. The second dashboard presents the aircraft clusters with a 3D scatter plot. The third dashboard plots the distinctive 3D coverage areas of aircraft clusters, encompassing all aircraft belonging to each specific cluster. Moreover, the recommendations for the clusters for their core network selection are given.

## 3 PERFORMANCE EVALUATION

We evaluate the Real-Time Digital Twin Platform by using the latency, clustering accuracy and cluster change rate metrics. Here, the latency is categorized as the preprocessing latency and the decision latency, corresponding to the time taken for the data integration and the decision-making in the recommendation engine. To withstand the dynamic of the environment, the platform should operate with low latency. To evaluate that, we measure the decision latency of our implementation, as shown in Figure 2. Here, as the number of aircraft increases, the delay also increases due to the processed data volume. Moreover, while the latency incurred from just the decision-making process shows similar results to the initial study [12], we observe that the preprocessing of the data mainly dominates the total latency. This is because, as the number of aircraft increases, the data retrieved from the API gets larger linearly. Correspondingly, it is processed for use in clustering. After the cluster decisions are made, the recommendation engine only processes one data point for each aircraft, making the decision latency comparably small to the preprocessing latency. This also showcases the importance of designing an efficient data pipeline.

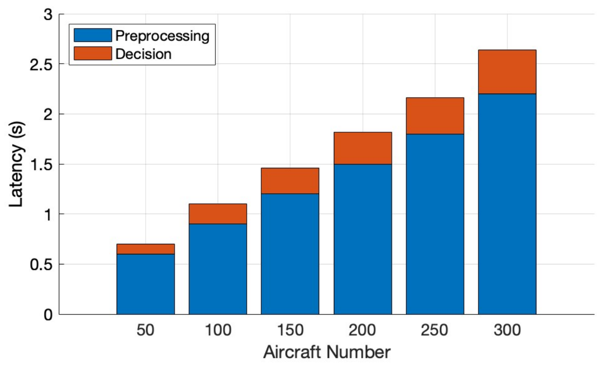


**Fig. 2** Evaluation of the latency for preprocessing and decision making.

Moreover, in Figure 3, the accuracy of the cluster assignment when the projection is applied is shown with the corresponding cluster change rate. Here, the clustering accuracy is calculated by using the projection step for an instance of extracted data and comparing the results for the next extraction step. As for the cluster change rate, it is the

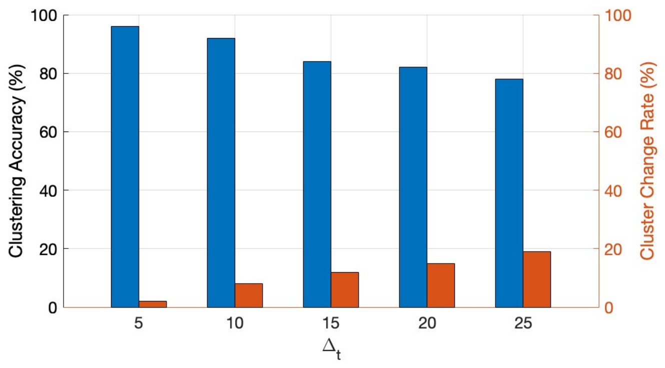


**Fig. 3** Evaluation of the data preprocessing pipeline for decision making.

percentage of cluster assignment that has to be changed compared to the baseline. In the core network selection, the cluster change occurs to provide better connectivity for the aircraft, and the projection is applied to ensure that the necessary cluster change decision can be made if the data cannot be retrieved. Here, the results show that the platform can make these decisions thanks to this robust mechanism. However, it is also observed that, as the $\Delta t$ increases, the accuracy of the results lowers, which is due to Equations 1-3 not considering the aircraft's acceleration and the emergence of the new aircraft. Meanwhile, using projection in further time differences causes a higher cluster change rate. This shows that the wrong cluster assignments affect the decision-making process's effectiveness.

## 4 CONCLUSION AND FUTURE WORKS

In this article, we implement a real-time digital twin platform for core network selection in aeronautical ad-hoc networks. This platform is built on a microservice-based architecture with three layers: Physical, DT, and Virtual Layers. Moreover, we design a robust data pipeline to integrate real-time aircraft data, handle the missing data, and decide on the core network selection. Then, we develop an interactive user interface using open-source tools. Correspondingly, we analyze the results for latency, accuracy of cluster assignment and cluster change rate. The results showcased that the proposed platform can integrate real-time data and apply decision-making with delays lower than 3 seconds. In this analysis, we revealed that preprocessing operations mainly dominate the system's latency. Considering this latency, we employed a projection step which ensures that the necessary cluster change decisions can be made in time. In future work, we plan to investigate using artificial intelligence (AI) / machine learning (ML) based methodologies at the projection step to improve the clustering accuracy and reduce the cluster change rate.

## ACKNOWLEDGEMENTS

This work is supported by The Scientific and Technological Research Council of Turkey (TUBITAK) 1515 Frontier R&D Laboratories Support Program for BTS Advanced AI Hub: BTS Autonomous Networks and Data Innovation Lab. Project 5239903.

## REFERENCES

[1] D. Lehner, J. Pfeiffer, E.-F. Tinsel, *et al.*, "Digital twin platforms: Requirements, capabilities, and future prospects," *IEEE Software*, vol. 39, no. 2, pp. 53–61, 2022. DOI: 10.1109/MS.2021.3133795.

[2] A. Fuller, Z. Fan, C. Day, and C. Barlow, "Digital twin: Enabling technologies, challenges and open research," *IEEE Access*, vol. 8, pp. 108 952–108 971, 2020. DOI: 10.1109/ACCESS.2020.2998358.

[3] K. Duran, M. Özdem, T. Hoang, T. Q. Duong, and B. Canberk, "Age of twin (aot): A new digital twin qualifier for 6g ecosystem," *IEEE Internet of Things Magazine*, vol. 6, no. 4, pp. 138–143, 2023. DOI: 10.1109/IOTM.001.2300113.

[4] E. Ak and B. Canberk, "Fsc: Two-scale ai-driven fair sensitivity control for 802.11ax networks," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6. DOI: 10.1109/GLOBECOM42002.2020.9322153.

[5] L. V. Cakir, K. Duran, C. Thomson, M. Broadbent, and B. Canberk, "Ai in energy digital twining: A reinforcement learning-based adaptive digital twin model for green cities," in *ICC 2024 - IEEE International Conference on Communications*, 2024, pp. 4767–4772. DOI: 10.1109/ICC51166.2024.10622773.

[6] D. M. Gutierrez-Estevez, B. Canberk, and I. F. Akyildiz, "Spatio-temporal estimation for interference management in femtocell networks," in *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications - (PIMRC)*, 2012, pp. 1137–1142. DOI: 10.1109/PIMRC.2012.6362517.

[7] B. R. Barricelli, E. Casiraghi, and D. Fogli, "A survey on digital twin: Definitions, characteristics, applications, and design implications," *IEEE Access*, vol. 7, pp. 167 653–167 671, 2019. DOI: 10.1109/ACCESS.2019.2953499.

[8] E. Ak, K. Duran, O. A. Dobre, T. Q. Duong, and B. Canberk, "T6conf: Digital twin networking framework for ipv6-enabled net-zero smart cities," *IEEE Communications Magazine*, vol. 61, no. 3, pp. 36–42, 2023. DOI: 10.1109/MCOM.003.2200315.

[9] Y. Wu, K. Zhang, and Y. Zhang, "Digital twin networks: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 13 789–13 804, 2021. DOI: 10.1109/JIOT.2021.3079510.

[10] Y. Yigit, B. Bal, A. Karameseoglu, T. Q. Duong, and B. Canberk, "Digital twin-enabled intelligent ddos detection mechanism for autonomous core networks," *IEEE Communications Standards Magazine*, vol. 6, no. 3, pp. 38–44, 2022. DOI: 10.1109/MCOMSTD.0001.2100022.

[11] Inmarsat Aviation, *Inflight connectivity survey – global whitepaper i august 2018*. [Online]. Available: https://www.inmarsat.com/content/dam/inmarsat/corporate/documents/aviation/insights/2018/Inmarsat%20Aviation%202018%20Inflight%20Connectivity%20Survey%20ENG.pdf.

[12] T. Bilen, E. Ak, B. Bal, and B. Canberk, "A proof of concept on digital twin-controlled wifi core network selection for in-flight connectivity," *IEEE Communications Standards Magazine*, vol. 6, no. 3, pp. 60–68, 2022. DOI: 10.1109/MCOMSTD.0001.2100103.

[13] Flightradar, *Live flight tracker - real-time flight tracker map*, en. [Online]. Available: http://FlightRadar24.com.

[14] C. Zhou, H. Yang, X. Duan, *et al.*, *Network digital twin: Concepts and reference architecture*, en. [Online]. Available: https://datatracker.ietf.org/doc/draft-irtf-nmrg-network-digital-twin-arch/.

[15] F. TAO, X. SUN, J. CHENG, *et al.*, "Maketwin: A reference architecture for digital twin software platform," *Chinese Journal of Aeronautics*, vol. 37, no. 1, pp. 1–18, 2024, ISSN: 1000-9361. DOI: https://doi.org/10.1016/j.cja.2023.05.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1000936123001541.

[16] M. Redeker, J. N. Weskamp, B. Rössl, and F. Pethig, "Towards a digital twin platform for industrie 4.0," in *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*, 2021, pp. 39–46. DOI: 10.1109/ICPS49255.2021.9468204.

[17] G. Blinowski, A. Ojdowska, and A. Przybyłek, "Monolithic vs. microservice architecture: A performance and scalability evaluation," *IEEE Access*, vol. 10, pp. 20 357–20 374, 2022. DOI: 10.1109/ACCESS.2022.3152803.

[18] *Opensky rest api*, https://openskynetwork.github.io/opensky-api/rest.html, Accessed: 2023-05-19, 2023.

[19] M. Schafer, M. Strohmeier, V. Lenders, I. Martinovic, and M. Wilhelm, "Bringing up opensky: A large-scale ads-b sensor network for research," in *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, 2014. DOI: 10.1109/IPSN.2014.6846743.

[20] *Get started with influxdb oss 2.7*, https://docs.influxdata.com/influxdb/v2.7/, Accessed: May 21, 2023.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[22] E. Schubert, "Stop using the elbow criterion for k-means and how to choose the number of clusters instead," *ACM SIGKDD Explorations Newsletter*, vol. 25, no. 1, pp. 36–42, Jun. 2023. DOI: 10.1145/3606274.3606278.

[23] M. Grinberg, *Flask Web Development: Developing Web Applications with Python*. O'Reilly Media, Inc., 2018.

[24] Grafana Labs, *Grafana documentation*, https://grafana.com/docs/, Online; accessed July 25, 2019, Jul. 2019.

[25] Plotly Technologies Inc., *Collaborative data science*, 2015. [Online]. Available: https://plot.ly.

ITU/JWCC

# Network Optimizing Software Solution for Multiplayer Gaming

**Arda Deniz Çelik**[1], **Gökhan Seçinti**[1]

[1] Department of Computer Engineering, Istanbul Technical University, Istanbul, 34469, Turkey

**Abstract:** In the digital entertainment landscape, multiplayer online games have gained immense popularity, attracting millions globally. This paper focuses on improving gaming performance by addressing two major challenges: latency and packet loss. Utilizing the Omnet++ application with the INET framework, we propose innovative network solutions to enhance the overall gaming experience. Over the past decade, multiplayer games have rapidly gained popularity due to technological advancements. As living costs rise, these games have become an economical and accessible form of entertainment, especially among the youth. The growing importance of a stable, high-quality internet connection in gaming has become crucial. Game developers and online gaming platforms also stand to gain from our network optimization solutions. Implementing these advancements can elevate service quality, leading to increased user satisfaction, positive reviews, and a stronger reputation in the gaming community. Additionally, this paper explores using real-world gaming data to simulate and test network strategies in Omnet++, offering practical insights for enhancing real-time online gaming experiences.

**Keywords:** End-to-End game latency, real-time packet loss, latency-sensitive network protocol, core network QoS for gaming.

## Çok Oyunculu Oyun için Ağ Optimizasyon Yazılımı Çözümü

**Özet:** Dijital eğlence dünyasında, çok oyunculu çevrimiçi oyunlar büyük popülerlik kazanmış ve dünya genelinde milyonları kendine çekmiştir. Bu proje, oyun performansını iyileştirmek için gecikme ve paket kaybı gibi iki ana zorluğu ele almayı hedeflemektedir. Omnet++ uygulaması ve INET çerçevesi kullanılarak, genel oyun deneyimini geliştirecek yenilikçi ağ çözümleri öneriyoruz. Son on yılda, çok oyunculu oyunlar teknolojik ilerlemeler sayesinde hızla yaygınlaşmıştır. Artan yaşam maliyetleriyle birlikte, bu oyunlar özellikle gençler arasında ekonomik ve erişilebilir bir eğlence biçimi haline gelmiştir. İstikrarlı ve yüksek kaliteli bir internet bağlantısının önemi bu süreçte daha da belirgin hale gelmiştir. Oyun geliştiricileri ve çevrimiçi oyun platformları da ağ optimizasyon çözümlerimizden faydalanmaktadır. Bu gelişmeler, hizmet kalitesini artırarak kullanıcı memnuniyeti ve olumlu geri bildirimlerle daha güçlü bir itibar kazandırabilir. Ayrıca, proje kapsamında Omnet++'da gerçek oyun verileri kullanılarak yapılan simülasyonlar, gerçek zamanlı çevrimiçi oyun deneyimlerini iyileştirmeye yönelik pratik bilgiler sunmaktadır.

**Anahtar Kelimeler:** Uçtan uca oyun gecikmesi, gerçek zamanlı paket kaybı, gecikmeye duyarlı ağ protokolü,oyun içi çekirdek ağ kalite hizmeti.

RESEARCH PAPER

Corresponding Author: Gökhan Seçinti, secinti@itu.edu.tr

Reference: A. D. Çelik, G. Seçinti, (2024), "Network Optimizing Software Solution for Multiplayer Gaming," *ITU Journ. Wireless Comm. Cyber.*, 1, (1) 47–56.

Submission Date: Aug, 27, 2024
Acceptance Date: Sept, 26, 2024
Online Publishing: Sept, 30, 2024

## 1 INTRODUCTION

It is the digital world that has revolutionized online multiplayer gaming, engaging millions of players across the globe in different platforms. With these games evolving, however, the demand for improved network performance is fast becoming an imperative. Among the most critical challenges, latency and packet loss both affect the quality of experiences; in an environment filled with competition or a league, decisions are made in a split second and matter a lot. Addressing these issues is important for improving the user experience and ensuring fair gameplay; this is particularly a need in the professional eSports context.

However, existing work has been focused on the performance of gaming and the relation between network characteristics like latency, packet loss, and player behavior. For instance, the effect of network quality on player exit behavior has been investigated by several studies, which indicate that poor network conditions may cause early player churn from games. Other works have investigated different network architectures and protocols, with a clear emphasis on genre-specific solutions that cater to the very specific demands of different classes of games. Such works represent the importance of network performance optimization for an improvement in gaming experiences across a different genre. Building on this work, our study implements and further develops network optimization approaches within the Omnet++ simulation environment, focusing on low-latency jitter handling in a multiplayer gaming context. By exploiting real gaming-world data using advanced simulation techniques, we hope to give practical answers that will not only enhance network performance but also be beneficial for the greater research into online gaming.

The major contributions of this paper are as follows:

i. A comprehensive analysis of network attributes affecting latency and packet loss in multiplayer games.

ii. Development and implementation of optimized network strategies within the Omnet++ environment.

iii. Evaluation of these strategies' impact on game performance, particularly in reducing latency and packet loss, thereby enhancing the overall gaming experience.

The rest of the paper is organized as follows: **Section II** summarizes the related work. Next, **Section III** presents the system model and methodology. **Section IV** showcases simulation setup and the performance evaluation. Lastly, **Section V** concludes the paper and discusses potential future work.

## 2 RELATED WORKS

In some studies which are about [1], [2], [3] FPS game traffic analysis, extensive research was conducted to investigate the influence of internet latency on game performance. Utilizing tools such as the Ethereal packet sniffer and Ns2 simulation, the study examined how various network attributes impact latency by analysing inter-send and inter-arrival times through bandwidth analysis on a clean and realistic client within the Quake III game [1]. In addition of the referenced study that primarily focused on analyzing traffic characteristics, the approach taken involves the active modification of the network environment to enhance real-time gaming experiences. This extension of research is indicative of a more practical application of network performance optimization in gaming scenarios.

Some researchers Ben, Youry and co-authors have used the TechEmpower company's 12-step test procedure to examine in detail what kind of delay data different web frameworks encounter after the benchmark. To analyze the results of these tests, they examined almost 400,000 gaming sessions using a web application called Gperf2 Collector [4]. The paper shares this focus on latency but differs in methodology. While the study relied on a virtualized network for understanding latency effects, paper uses real-world gaming data in Omnet++ simulations. This approach allows to directly apply network theory to practical gaming situations, offering more tailored solutions for improving gaming network performance.

The comprehensive analysis of Netcode in online multiplayer games, focusing on aspects such as ping, server tick rate, and the extent of delay in game dynamics for each game, resonates with the objectives of the paper. This field is advanced by the work undertaken, which not only involves analyzing network issues but also seeks to mitigate them through network optimization strategies [5]. The integration of tools such as Wireshark and Ping-Plotter, as outlined in the study, is a fundamental aspect of this paper. This highlights the practical application and relevance of these tools in addressing real-world network optimization challenges.

In the study "Client-Side Network Delay Compensation for Online Shooting Games, the authors specifically focused on synchronization and unfair gameplay in multiplayer shooting games [6]. The article suggests that current server load balancing methods are insufficient and to solve this, it recommends reducing the data used and estimating the player's location by using a regression-based method. As a result of their tests and evaluations, they claimed that the solution method they proposed provided an improvement of nearly 16 pixels in estimating player location. By using this method, it has made a significant contribution to the development and reduction of the queuing time of this paper on Omnet++.

In study "A Multiplayer Real-Time Game Protocol Architecture for Reducing Network Latency" introduces an original client/server architecture for multiplayer real-time video games to reduce network latency [7]. An STU segment queue approach serializes sporadic events into periodic groups for scheduling efficiency, and a spatial domain approach selects clients affected, updating only those clients to reduce the data transmission. The implemented architecture in Java, with non-blocking I/O and thread pool management, makes servers more efficient, reusing resources to increase the throughput of garbage collection. Our proposed architecture's experimental results show clearly reduced latency and synchronization very well for real-time gaming environments with high-quality service. These results have implications for our paper and enable approaches that reduce latency and increase synchronization and can be integrated into our system for better scaling.

Petlund study focuses on [8] the challenges of reducing latency in interactive applications, particularly online multiplayer games, that generate "thin streams." In sessions of ”thin stream” interactive applications, especially multiplayer games, traditional TCP mechanics decrease performance because packets with small payload and high interarrival are sent as a stream on the network. It is emphasized that congestion control and recovery mechanisms, which are features of TCP, are almost never triggered due to the inadequacy of the sent packets. In order to improve the TCP protocol, which is inadequate due to these packet features, the author, who focused on the game where 30-byte packets called BZFlag were sent and the packet interarrival time was 24ms, observed that the delays in the application and transport layers were significantly reduced by removing the exponential backoff and a few different modifications to reduce the delay experienced by the player. The study conducted in this paper, it has been revealed that although UDP is more difficult to create and less efficient in terms of security, it is the protocol that provides less delay in games thanks to its customizability and the simplicity of the mechanisms within it.

Many studies [2] [9] focuses on which network architecture and network design should be chosen for game genre specific because they found out that there is no superior architecture for every game genre is In Moll's study [2], they focused on the problems of network inadequacy in modern multiplayer games, especially in the Battle Royale genre. Unlike the traditional network protocols used in these games, they proposed to use Information-Centric Networking (ICN) and Named Data Networking (NDN) to develop network protocols with the data they obtained during the game sessions of nearly 36 hours in order to meet the low latency requirement in competitive games. Thanks to this proposal, they managed to significantly reduce network latency and optimize bandwidth usage. The research conducted contributed to the evaluation of possible network solutions by providing data for the test environments created in the creation of this paper.

In the study conducted by Chen [10], they tested the effect of network quality on Massively Multiplayer Online Role-Playing Games. Based on 1.356 million packet data, they observed that players left the game prematurely than their average game time due to packet loss and network delays. When the logistic regression model was applied, the study quantifies player "intolerance" to various network impairments, finding that the degrees of player intolerance to network delay, delay jitter, client packet loss, and server packet loss are approximately in the proportion of 1:2:4:3. In addition, since packet loss is the network variable that causes the most player loss, it was observed that directing players with poor internet connection to better quality servers increases the average game time overall. Thanks to this study, it played an important role in determining our main variable target as packet loss in the research conducted in this paper.

In the study [11], authors mention that they conducted a comprehensive study on the effects of network delays on the gaming experience for end users. They have shown that many different parameters such as game type, environmental factors and distance to servers are important for delays in games. For the tests carried out in this study, they created an analysis by using real-time data received through the WTFast GPN application and passing it through machine learning, which determines the game sessions at different latency levels they developed with high precision. As a result of this analysis, it was revealed that network delay is the most important factor in gaming experience and it was revealed that the machine learning model with the Random Forest algorithm was the most effective model in determining game delays among 8 different machine learning models for reducing network delay. This study has shown that machine learning can be an important supporting factor in developing gaming experience.

In the study [12], the author has found a study using game theory on dynamic resource sharing for production networks with stochastic demand. He also stated that production systems, especially those using cloud systems, are the basis for creating this simulation. With the solution presented by the author, it has been shown that, unlike traditional models, namely dedicated and full-flexibility systems, the Gale-Shapley algorithm plays a much more effective role in the trade-off between performance and the complexity caused by active links in previous studies. She also proposed that a fast computational process (scaling at $n\hat{2}$ proposals), making it ideal for real-time reconfiguration in production environments. Thanks to the solution proposed by the author, it has also been stated that it is as reliable as traditional models against demands in changing conditions and has a lower network complexity.

## 3 SYSTEM MODEL

### 3.1 Omnet++ with INET Framework

Omnet++ is a network simulation platform used to create and analyze a realistic multiplayer gaming environment. The INET framework was integrated with Omnet++ to model various network components such as hosts, routers, switches, and game servers. This setup enabled detailed simulations of network behavior under different conditions, with a focus on minimizing latency and reducing packet loss.

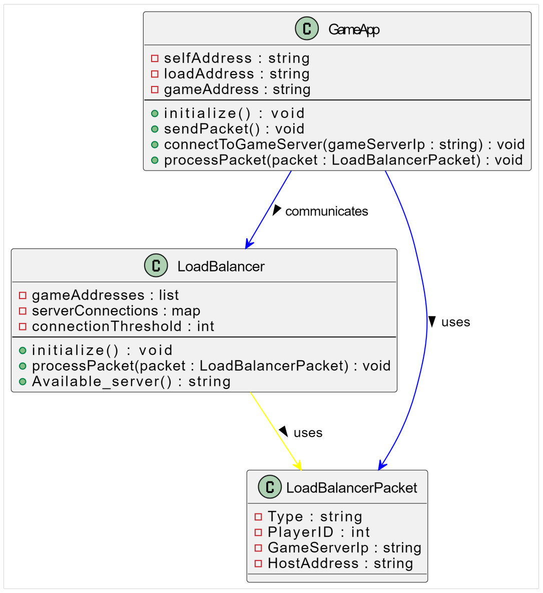


**Fig. 1** Class diagram of solution.

### 3.1.1 Structural model for GameApp

In our multiplayer gaming simulation, **GameApp** is designed to handle interactions that are effective on the client side and with both the loadserver and the game server. This model has many important roles such as managing network connections, ensuring the effective distribution of game data, and processing the data received by the loadserver.Here's a detailed breakdown of the structural model of the **GameApp** module, covering its key algorithms and functional flows:

#### Initialization and Connection Setup

The initialization of the **GameApp** starts with the `processStart()` function. This function is responsible for setting up network sockets and initializing the connection settings.

```
procedure processStart
    initializeSocket()
    loadAddress <- getLoadBalancerAddress()
    connectLoadServer()
end procedure
```

This function determines the socket options for the host using GameApp to connect to the loadserver, and also ensures that the host's own address and loadserver adress to be sent to the loadserver is kept as an L3Address variable so that it can be sent. It then calls the connectLoadServer function to send the first packet and obtain the IP address of the appropriate game server.

#### Connecting to the Load Server

Once the initial setup is complete, the `connectLoadServer()` function is called. This function creates and sends a packet to the load balancer to determine an appropriate game server for the client to connect to, based on current server loads.

```
procedure connectLoadServer
    packet <- createPacket("Connection Request")
    sendPacket(loadAddress, packet)
end procedure
```

In this function, the first package is created and sent in order to establish a connection to the loadserver. Additionally, it is first checked whether there is any fragmentation error, and after function set the essential and custom variables to proper values to ensure communication between network layers, function send packet to loadserver using loadAdress which is stored in processStart function.

#### Processing Packets from the LoadBalancer

The `processPacket()` function takes over upon receiving a packet from the load balancer. This function processes different types of packets based on their content, primarily focusing on handling server assignment packets.

```
procedure processPacket(packet)
    switch packet.type
        case "Server Assignment":
            gameServerIp <- packet.getGameServerIp()
            destAddress.push_back(gameServerIp)
            processSend()
    end switch
end procedure
```

After the packet sent by the loadserver arrives, the function checks the type of the incoming packet and it is analyzed by the loadserver module that the incoming packet was sent in the correct packet type. Then, using the gameServerAdress payload sent in the package, the destination address required for the host to connect to the game is determined. Finally, the processSend function is called.

#### Sending Data to the Game Server

The `processSend()` function uses the inherited functionalities from `UdpBasicApp` to handle the actual data transmission to the assigned game server.

```
procedure processSend
    sendPacket(gameServerIp, createGamePacket())
end procedure
```

The ProcessSend() function creates and sends the first package whose destination address has been determined by processpacket and is ready to connect with the game server. While doing this, it uses the function inherited by UdpBasicApp.

### Detailed Packet Handling

The `sendPacket()` function constructs a packet with the appropriate type and content, then sends it to the designated server address. This is crucial for game state updates and player actions.

```
procedure sendPacket(address, data)
    packet <- new Packet(data)
    packet.setType("APP_SERVER")
    network.sendTo(address, packet)
end procedure
```

This function ensures that each packet is correctly formatted and dispatched to maintain continuous and real-time gameplay interaction.

All functions in the **GameApp** application are a whole that allows the hosts to find a suitable game server through the loadserver and to process the packet received by the loadserver and to be in constant communication with the game server.

### 3.1.2 Structural model for LoadBalancer

The **LoadBalancer** module within our OMNeT++ simulation plays a critical role in managing server loads and distributing client requests across multiple servers. Its design ensures efficient processing of network connections and server assignments, pivotal for maintaining an optimal performance environment in multiplayer gaming scenarios. Below is a detailed explanation of the structural model of the **LoadBalancer** module, including key algorithms and pseudocode implementations.

### Initialization Process

The initialization of the **LoadBalancer** leverages inheritance from the `UdpBasicApp` to set up initial module configurations.

```
procedure Initialize
    UdpBasicApp.Initialize()
    setupInitialConfiguration()
end procedure
```

This function calls the initialization method of the inherited `UdpBasicApp`, ensuring that all underlying network functionalities are correctly set up.

### Starting the Process

The `processStart()` function is crucial for preparing the LoadBalancer to handle incoming connections and manage server addresses effectively.

```
procedure processStart
    bindSocket()
    for each address in initialGameAddresses
        gameAddressStr.push_back()
    end for
end procedure
```

After first determining the sockets for the loadserver, the ProcessStart() function makes preliminary adjustments to find suitable game servers by pushing the game server addresses given as parameters in the omnetpp.ini file to an L3Adress vector and defining an integer variable for each IP address.

### Handling Messages

The `handleMessageWhenUp()` function utilizes the capabilities inherited from `UdpBasicApp` to manage incoming network messages.

```
procedure handleMessageWhenUp(message)
    super.processMessage(message)
end procedure
```

This function ensures that messages either from upper or lower network layers are processed according to the base app's logic.

### Server Availability Check

The `Available_server()` function determines the optimal server for new connections based on current load conditions.

Iterates through the entire list of game server addresses, checking whether each function has a connection count greater or less than the connectionThreshold integer value. If there is an IP address smaller than connectionThreshold, this IP address is returned as the function result. If there is not one, the threshold value is overridden and reassigned to the servers.

### Packet Processing

The `processPacket()` function is tasked with handling incoming packets from hosts, determining the available server, and responding appropriately.

```
procedure processPacket(packet)
    serverIp <- Available_server()
    responsePacket <- createPacket(serverIp)
    sendPacket(packet.origin, responsePacket)
    updateConnectionCount(serverIp, increment)
end procedure
```

The ProcessPacket() function first checks the type of the incoming package. If the package is a package sent by one of the host to loadserver, the host address is extracted from the incoming package and given to the newly created package as the destination address. In order for the payload to be sent correctly, variables are set and the L3 game server address sent by the Available_server() function is added to the package and the package is sent to the host. Finally, the connection number of the game server address sent to connect the host is updated to +1.

#### Connection Count Management

Lastly, the `updateConnectionCount()` function adjusts the count of current connections per server, ensuring accurate load tracking.

```
procedure updateConnectionCount(serverIp)
  if increment
      serverConnections[serverIp]++
  else
      serverConnections[serverIp]--
  end if
end procedure
```

This function modifies the connection count based on whether a new connection is being added or an existing one is terminated.

With the help of all the functions in the Loadbalancer application, game server addresses are kept through a map in the loadserver, incoming packages are processed and sent to their new destinations with the appropriate game server addresses. This is the most important of the improvements made to improve the gaming experience.

### 3.1.3 Structural model for LoadBalancer packet

The **LoadBalancerPacket** plays a crucial role in the communication framework of our networked multiplayer game environment, acting as the primary data carrier between the **GameApp** and **LoadBalancer** modules. This packet is designed to encapsulate all necessary data for effective load balancing and server assignment. Below, we detail the structural model of the **LoadBalancerPacket**, including its design and key functionalities.

#### Packet Type Enumeration

The packet types are defined by an enumeration `APP_TYPE`, which specifies the kinds of packets that can be generated within the system. This helps in distinguishing the purpose of each packet as it flows through the network.

```
enum APP_TYPE {
    CONNECTION_REQUEST,
    SERVER_ASSIGNMENT,
    DATA_TRANSFER
}
```

This enumeration facilitates the handling of packets based on their type, improving the routing and processing efficiency within the network.

#### Class Definition and Attributes

The class `LoadBalancerPacket` is designed with attributes that support comprehensive data encapsulation for network operations.

```
class LoadBalancerPacket {
    APP_TYPE type
    int playerID
    string gameServerIp
    string hostAddress
}
```

Here are the custom parameters of the `LoadBalancerPacket`: - `type`: An instance of `APP_TYPE`, determining the packet's function within the network. - `playerID`: An identifier for the player sending the request, aiding in response management and data tracking. - `gameServerIp`: The IP address of the assigned game server, used in server assignment operations. - `hostAddress`: The originating address of the packet, typically used for routing back responses or data.

### 3.2 Netlimiter

NetLimiter was employed to monitor and manage network traffic during gaming sessions. By analyzing download and upload rates for each application, NetLimiter provided essential insights into how the network behaves in real-time. The IP filtering feature allowed the study to focus on specific network interactions, particularly those related to the game servers, enabling a targeted approach to traffic analysis.

### 3.3 Wireshark

Wireshark, a network protocol analyzer, was used to capture and examine packet data flowing through the network during gaming sessions. It provided detailed information on packet sizes, transmission intervals, and overall traffic patterns. This data was instrumental in refining the simulation model within Omnet++, ensuring that it accurately reflected real-world network conditions. Wireshark's ability to filter and analyze traffic based on specific IP addresses, identified through NetLimiter, made it a vital tool in the study's methodology.

## 4 PERFORMANCE EVALUATION

The network design where the main tests of the paper are carried out is seen in the Figure. This figure consists of both 2 different game servers and hosts that want to play more than one game at the same time to create and test a realistic game environment. In order to connect to the game servers, the hosts first pass through their local switches to the routers and then through the ISP router to reach the servers. The important aspect of the design is that instead of the hosts connecting directly to the servers, a load balancer server is created that decides which servers are more suitable for the hosts to connect to. In this network design, thanks to the customized packages in the load balancer, the connection availability statuses determined by the instant occupancy rates of the servers are kept in the load balancer server and it allows the host to direct the incoming game connection request to the most suitable game server for itself. Additionally load balancer made the best game server selection is decided for each user in a way that allows players to benefit from less packet loss and delay.

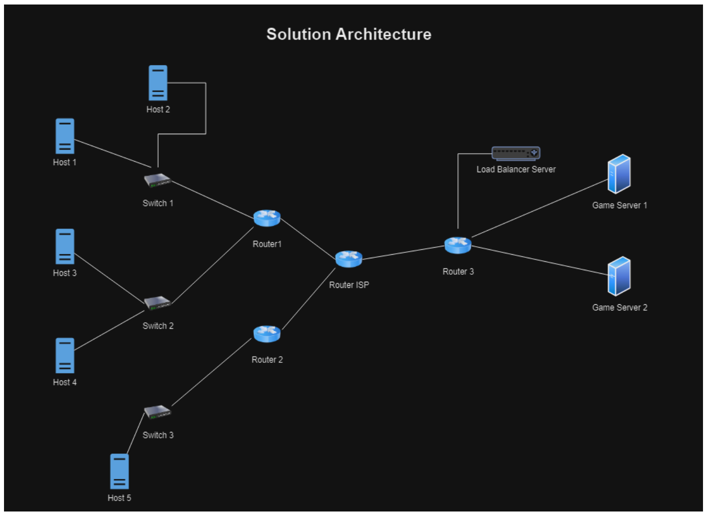


**Fig. 2** Solution design in Omnet++.

In the study carried out using Ethereal packet sniffer and Ns2 simulation on behalf of FPS games, [1] focused only on the Quake III game, examined the network features and traffic structure of this game, and carried out studies to improve the gaming experience by making modifications, especially in terms of throughput and delay. In this study, the game traffic of more than one game was examined using NetLimiter and WireShark applications to ensure less delay and throughput. As a result of work have done to prevent queuing time and possible delays that may occur, even if only in small numbers,the packet loss was successfully reduced to a value very close to zero, effectively preventing potential instant delays, as demonstrated in Figure 3. To achieve this by adding a slight delay and allowing each game server to send packets one by one, instead of constantly bombarding the game servers with packets at the same time. This enabled to achieve the goal of processing 1000 operations per second that.

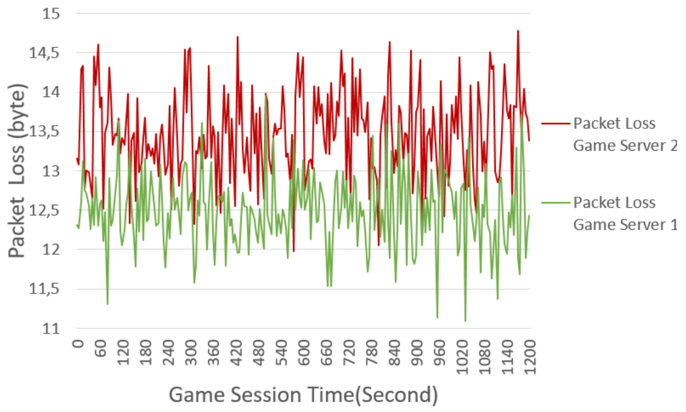


**Fig. 3** Packet loss of game servers.

In [4] study, an average of 400,000 gaming sessions are examined using TechPower company procedures. The web application called Gperf2, which they tested in this study, inspired this paper to develop the queuing time of the model. By using the approach of this team, which also tested the problems that may be caused by overloading the server with the techniques they used, the queuing time value shown in Figure 4 is reached for game server 1 and game server 2, which shows significant improvement with the use of Load balancer rather then without using the Load balancer module. In this way, queuing time development is completed to reduce instant spikes in queueing time. Use of Load balancer was increased the sharpness of the sended packet response time and improve the overall quality of the game for players. Additionally, with the reduced queueing time for servers, overall player play time will be increase significantly due to more reliable conenction status.

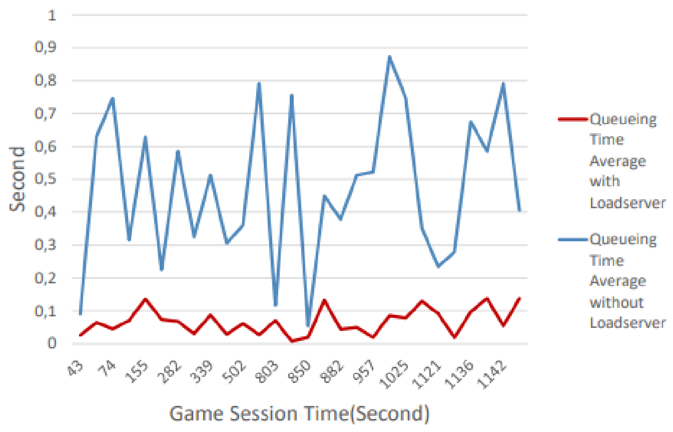


**Fig. 4** Loadserver queueing time.

The goal of reducing the observed delay between 5% and 7% and packet lose by 8%, the research paper [5] inspired this work to achieve the goal. The main purpose of this research, which makes a comparative Netcode analysis of online multiplayer games, is to examine the essential data of the game experience such as ping and server tick rate and find solutions to the problems that occur. Using applications such as WireShark and Ping-Plotter to find these solutions, the research team developed different implementation strategies for servers and included them in their research to solve the delays and packet losses that occur. In this study, thanks to the LoadBalancer and GameApp applications, both the delays that may occur on the servers and the packet losses were improved by 6.35% and overall delay reduce by 7%. In other Figure 5 represent a essential data for server. Figure give a insight about upcoming packages are passed to servers without noticable losses which is a big problem in the start phase of the paper.

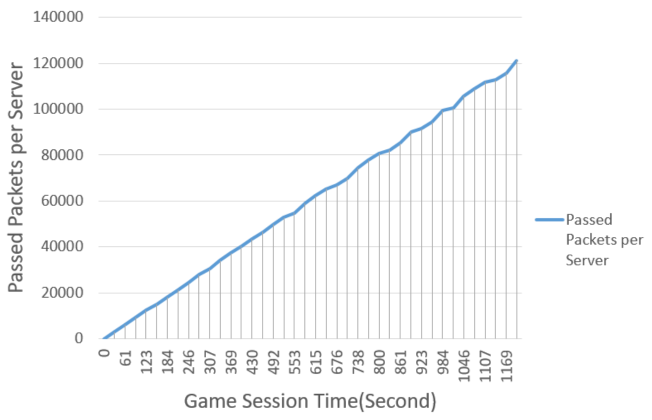


**Fig. 5** Passed packet per server.

## 5 CONCLUSION AND FUTURE WORK

### 5.1 Conclusion

In this paper, we developed and evaluated a network optimization solution for multiplayer gaming environments using the OMNeT++ simulation platform. The system demonstrated improved performance, achieving a throughput exceeding 1000 operations per second, reducing latency by approximately 7%, and decreasing packet loss rates by 6.35%. These improvements contribute to a more stable and enjoyable gaming experience.

### 5.2 Future Work

Future research can focus on several areas to further enhance the system:

- **Scalability Improvements**: Optimizing the LoadBalancer algorithm to efficiently manage increased traffic using more advanced algorithms.
- **Machine Learning for Load Balancing**: Implementing machine learning algorithms to dynamically adjust server allocations based on real-time network conditions.
- **Enhanced Security Measures**: Integrating advanced security protocols to protect against potential threats such as DDoS attacks.
- **Server Distribution Algorithm**: Adapting the system for more flexible and advanced server Ip adress distribution.
- **Real-Time Analytics Dashboard**: Developing a dashboard to visualize key network performance metrics for proactive management.

## ACKNOWLEDGEMENTS

This work is supported by The Scientific and Technological Research Council of Turkey (TUBITAK) 1515 Frontier R&D Laboratories Support Program for BTS Advanced AI Hub: BTS Autonomous Networks and Data Innovation Lab. Project 5239903 and Istanbul Technical University, Department of Scientific Research Projects (MÇAP-2022-43828).

## REFERENCES

[1] Q. Zhou, C. J. Miller, and V. Bassilious, "First person shooter multiplayer game traffic analysis," in *Proceedings of the 2008 International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC)*, 2008. DOI: 10.1109/isorc.2008.28.

[2] P. Moll, M. Lux, S. Theuermann, and H. Hellwagner, "A network traffic and player movement model to improve networking for competitive online games," in *Proceedings of the 2018 16th Annual Workshop on Network and Systems Support for Games (NetGames)*, Jun. 2018. DOI: 10.1109/netgames.2018.8463390. [Online]. Available: https://doi.org/10.1109/netgames.2018.8463390.

[3] T. Henderson, "Latency and user behaviour on a multiplayer game server," in *Lecture Notes in Computer Science*, 2001, pp. 1–13. DOI: 10.1007/3-540-45546-9_1. [Online]. Available: https://doi.org/10.1007/3-540-45546-9_1.

[4] B. Ward, Y. Khmelevsky, G. Hains, R. Bartlett, A. Needham, and T. Sutherland, "Gaming network delays investigation and collection of very large-scale data sets," in *2017 Annual IEEE International Systems Conference (SysCon)*, Apr. 2017. DOI: 10.1109/syscon.2017.7934779.

[5] M. Ahmed, S. Reno, M. R. Rahman, and S. H. Rifat, “Analysis of netcode, latency, and packet-loss in online multiplayer games,” in *2022 International Conference on Augmented Intelligence and Sustainable Systems (ICAISS)*, Nov. 2022. DOI: 10.1109/icaiss55157.2022.10010926.

[6] T. Motoo, J. Kawasaki, T. Fujihashi, S. Saruwatari, and T. Watanabe, “Client-side network delay compensation for online shooting games,” *IEEE Access*, vol. 9, pp. 125 678–125 690, Jan. 2021. DOI: 10.1109/access.2021.3111180.

[7] Y. Ahn, A. K. Cheng, J. Baek, and P. Fisher, “A multiplayer real-time game protocol architecture for reducing network latency,” *IEEE Transactions on Consumer Electronics*, vol. 55, no. 4, pp. 1883–1889, Nov. 2009. DOI: 10.1109/tce.2009.5373746.

[8] A. Petlund, K. Evensen, P. Halvorsen, and C. Griwodz, “Improving application layer latency for reliable thin-stream game traffic,” in *Proceedings of the 7th ACM International Conference on Multimedia*, 2008. DOI: 10.1145/1517494.1517513. [Online]. Available: https://doi.org/10.1145/1517494.1517513.

[9] X. Che and B. Ip, “Packet-level traffic analysis of online games from the genre characteristics perspective,” *Journal of Network and Computer Applications*, vol. 35, no. 1, pp. 240–252, Jan. 2012. DOI: 10.1016/j.jnca.2011.08.005. [Online]. Available: https://doi.org/10.1016/j.jnca.2011.08.005.

[10] N. K.-T. Chen, P. Huang, and N. C.-L. Lei, “Effect of network quality on player departure behavior in online games,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 5, pp. 593–606, May 2009. DOI: 10.1109/tpds.2008.148. [Online]. Available: https://doi.org/10.1109/tpds.2008.148.

[11] A. Wong, C. Chiu, G. Hains, J. Behnke, Y. Khmelevsky, and T. Sutherland, “Network latency classification for computer games,” in *2021 IEEE Conference on Recent Advances in Systems Science and Engineering (RASSE)*, 2021. DOI: 10.1109/rasse53195.2021.9686848. [Online]. Available: https://doi.org/10.1109/rasse53195.2021.9686848.

[12] P. Renna, “Capacity and resource allocation in flexible production networks by a game theory model,” *The International Journal of Advanced Manufacturing Technology*, vol. 120, no. 7–8, pp. 4835–4848, Mar. 2022. DOI: 10.1007/s00170-022-09061-y. [Online]. Available: https://doi.org/10.1007/s00170-022-09061-y.