BİLGİSAYAR BİLİMLERİ VE TEKNOLOJİLERİ DERGİSİ

JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGIES

EDİTÖR: DOÇ. DR. Erdinç AVAROĞLU
ISSN 2717 - 8579

Bilgisayar Bilimleri ve Teknolojileri Dergisi

BİLGİSAYAR BİLİMLERİ VE TEKNOLOJİLERİ DERGİSİ

CİLT 4, SAYI 2

ISSN: 2717-8579

ARALIK 2023

Bilgisayar Bilimleri ve Teknolojileri Dergisi

Dergi Hakkında

Bilgisayar Bilimleri ve Teknolojileri Dergisi bilim ve teknolojideki gelişmelere paralel olarak bilgisayar bilimleri ve teknolojileri alanında yeni gelişmelerle ilgili yapılan çalışmaları yayınlayan bir dergidir.

Amaç & Kapsam

BIBTED Dergisi,

- Bilgisayar Bilimleri ve Teknolojileri Dergisinin amacı bilgisayar alanında yapılan özgün çalışmaları yayınlamaktır. Yazım kurallarına uygun olarak hazırlanan eser, dergi editörlüğünce değerlendirme için hakemlere gönderilir. Bilgisayar Bilimleri ve Teknolojileri Dergisinde **KÖR HAKEMLİK** uygulaması mevcuttur. Yayımlanmasına, hakemlerin görüşü doğrultusunda Dergi Editör ve Yayın Kurulu karar verir. Gönderilen makaleler yayınlansın veya yayınlanmasın iade edilmez. Dergimizde yayınlanan yazıların her türlü sorumluluğu (bilimsel, mesleki, hukuki, etik vb.) yazarlara aittir. Yayınlanan yazıların telif hakkı dergiye aittir ve referans gösterilmeden aktarılamaz. Araştırmacılar arasındaki bilimsel iletişimi oluşturmak amacıyla aşağıda nitelikleri açıklanan, başka bir yerde yayımlanmamış makaleler Türkçe ve İngilizce olarak kabul edilmekte ancak Türkçe Kabul edilen makalenin özetinin ingilizce de basılması zorunluluğu vardır.

Aşağıdaki türlerdeki makaleler dergide yayına kabul edilmektedir:

- **Araştırma makalesi**: Özgün bir araştırmayı sonuçlarıyla birlikte sunan makale,
- **Derleme makale:** Bilgisayar Mühendisliği alanında belli bir konuda yeterli sayıda bilimsel makaleyi tarayıp, özetleyen, değerlendirme yapan ve bulguları yorumlayan makale,
- **Endüstriyel makale:** Bu alanda endüstride yapılan araştırma ve geliştirilen yeni ürün veya teknolojilerin açıklandığı makale,
- **Tez çalışması:** Lisansüstü düzeyde yapılan özgün bir tez çalışmasının genişletilmiş özetini içeren yazı,
- **Kitap yorumu:** Bilgisayar mühendisliği alanında yayınlanmış yeni bir kitabın tanıtılması ve değerlendirilmesi.
- Kısa Bildiri: Yapılan bir araştırmanın önemli bulgularını açıklayan yeni bir yöntem veya teknik tanımlayan yazılar.

Bütün yazıların Telif Hakkı Devri, yazarlarına bir form gönderilmek suretiyle alınır. Telif Hakkı Devir Formu göndermeyen yazarların yayınları işleme konmaz. Yayınlanmasına karar verilen yazılar üzerine yazarlarınca hiçbir eklenti yapılamaz.

Her yazı konusu ile ilgili en az iki hakeme gönderilerek şekil ve içerik bakımından incelettirilir. Dergide yayınlanabilecek nitelikteki yazılar dizgisi yapıldıktan sonra, yazarlarına gönderilerek baskı öncesi gözden istenir. Makale içinde, dergide basıldığı haliyle gözüken hataların sorumluluğu yazarlarına aittir. Hata, editörlük ofisinden kaynaklandığı takdirde düzeltme yayınlanabilir.

Derginin Kapsamı;

Bilgisayar Bilimleri ve Teknolojileri Dergisinin kapsamı, akıllı sistemler, algoritmalar, benzetim, bilgisayar ağları, bilgisayar grafiği, bilgisayarla görme, bilgisayar mimarisi, bilgiye erişim, bilimsel hesaplama, bilişim güvenliği, biyoenformatik, kriptografi, paralel işleme, doğal dil işleme donanım, görüntü işleme, hesaplama kuramı, işaret işleme, işletim sistemleri, makine öğrenmesi, mobil sistemler, modelleme, tıbbi bilişim, veri madenciliği, veri tabanı sistemleri, yazılım mühendisliği, siber güvenlik, yapay zeka dahil olmak üzere bilgisayar bilimleri ve teknolojilerin tüm alanları içerir.

Yayınlanma Sıklığı Yılda 2 sayı

ISSN 2717-8579

WEB https://dergipark.org.tr/tr/pub/bibted

İletişim eavaroglu@mersin.edu.tr / ttuncer@firat.edu.tr / kemaladem@gmail.com

Bilgisayar Bilimleri ve Teknolojileri Dergisi

Bilgisayar Bilimleri ve Teknolojileri Dergisi

İçindekiler
Contents

ARAŞTIRMA MAKALELERİ;
RESEARCH ARTICLES;

BİLGİSAYAR BİLİMLERİ VE TEKNOLOJİLERİ DERGİSİ
JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGIES
http://dergipark.org.tr/tr/pub/bibted

Araştırma Makalesi

Makine Öğrenimi Yöntemlerini Kullanarak Kötü Amaçlı Yazılımların Statik Analiz ile Tespiti

Nisa VURAN SARI*[1], Mehmet ACI[2]

[1]Mersin Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği, Mersin, Türkiye
[2] Mersin Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği, Mersin, Türkiye

Anahtar Kelimeler:
Siber Güvenlik
Zararlı Yazılım Tespiti
Zararlı Yazılım Analizi
Makine Öğrenmesi

ÖZ

Siber saldırılardaki artış internet ve bilişim teknolojileri kullanımını da tehdit etmeye başlamıştır. Bu durum, siber saldırılardan sorumlu kötü amaçlı yazılımları tespit etmenin önemini vurgulamaktadır. Günümüzde, kötü amaçlı yazılımları algılamak için makine öğrenmesi yöntemlerinin geliştirilmesi üzerine çalışmalar bulunmaktadır. Kötü amaçlı yazılım dedektörleri, kötü amaçlı yazılımlara karşı savunmada birincil araçlardır. Böyle bir dedektöderün kalitesi, kullandığı tekniklerle belirlenir. Makine öğrenmesi, derin öğrenme ve statik ve dinamik analiz gibi zararlı yazılım analiz yöntemleri bu teknikler arasında yer almaktadır. Bu çalışma kötü amaçlı yazılım analizi ve sınıflandırma tekniklerini sunmaktadır. Kötü amaçlı yazılım tespiti için, K-En Yakın Komşular, Saf Bayes, Karar Ağaçları ve Rastgele Orman gibi iyi bilinen makine öğrenmesi algoritmaları kullanılmıştır. Çalışma, Karar Ağaçları sınıflandırma tekniği kullanımının %97,75 sınıflandırma ile en iyi doğruluğu ürettiğini, Saf Bayes'in ise %53 ile en düşük doğruluğu ürettiğini göstermektedir.

Detection of Malware by Static Analysis Using Machine Learning Methods

Keywords:
Cyber Security
Malware Detection
Malware Analysis
Machine Learning

ABSTRACT

The increase in cyber-attacks has also started to threaten the use of internet and information technologies. This situation emphasizes the importance of detecting malicious software that is responsible for cyber-attacks. Nowadays, there are studies on the development of machine learning methods for malicious software detection. Malicious software detectors are the primary tools in defense against malicious software. The quality of such a detector is determined by the techniques it uses. Malware analysis methods such as machine learning, deep learning, and static and dynamic analysis are among these techniques. This study presents malware analysis and classification techniques. For malware detection, well-known algorithms for machine learning including such K-Nearest Neighbors, Naive Bayes, Decision Trees, and Random Forest were used. The research shows that the use of Random Forest classification technique produces the best accuracy with 97.75% classification, while Naive Bayes produces the lowest accuracy of 53%.

***Sorumlu Yazar**

*(nvuran@mersin.edu.trl) ORCID ID 0000–0001–7042–3031
(maci@mersin.edu.tr) ORCID ID 0000–0002–7245–8673

e-ISSN: 2717-8579
Geliş Tarihi: 06/06/2023; Kabul Tarihi: 08/08/2023

1. INTRODUCTION

In recent years, there has been an increase in the development of malware. Today, cyber attackers use malware for attacks on information systems. Internet environments such as e-mails, malicious websites, downloaded software are the main environments for performing a malware attack on information systems. Detection and analysis of these malicious software is important for information security and healthy internet usage. In general, 3 basic techniques (static, dynamic and hybrid) are examined for the analysis of malicious software. Analyzes extract different features to classify and detect malicious and harmless files. Static and dynamic methods can be used to automate and speed up the steps in malware analysis, detection and classification. Extracting different malware features through analytical techniques is critical to the success of malware detection. Data from static or dynamic methods can be used to detect malware or classify malware by families using machine learning techniques (i.e., clustering or classification).

Analyzing malware without running it is called static analysis. Patterns used in static analysis include string signature, syntactic library call, byte array n-grams, opcode (operation code) frequency distribution, control flow diagram, and so on (Gandotra et al., 2014). Analysis of the behavior (interaction with the system) of a malware while running in a controlled environment such as a virtual machine, sandbox, simulator, emulator, and so on is called dynamic analysis. Before running the malware sample, appropriate monitoring tools such as Process Monitor, Capture BAT, Process Hacker, Process Explorer, Regshot and Wireshark are installed and enabled (Gandotra et al., 2014). Despite all its benefits, the biggest disadvantage of dynamic analysis is the performance cost. When dealing with large datasets containing hundreds of binaries, dynamic analysis cannot scale effectively (Hassen et al., 2017). Since scalability is a need of this research, static analysis is used to extract features from the malicious programs.

Although traditional malware classification methods were used before, with the rapid spread and complexity of malware, there is a need to develop new methods. Machine learning methods are also powerful technologies that transform the effectiveness of cyber security research. In this study, Random Forest (RF), Decision Tree (DT), K-Nearest Neighbor (KNN) and Gaussian Naive Bayes (NB) machine learning methods were used to classify malware. Calculated accuracy, f1-score, precision, and recall metrics are compared to determine success of the methods. Among the methods used, it is observed that the RF algorithm gave the most successful result with an accuracy rate of 97.75%.

Malware has posed a great threat to computer systems from past to present. New techniques and methods have been tested by adding new malware detection studies that have been carried out since the past. It has been observed that artificial intelligence-assisted methods such as machine learning and deep learning methods improve malware detections and outperform existing methods, with the efficiency of existing methods decreasing against new threats. In the continuation of this section, a summary of the studies on malware detection in the literature is presented.

Tian et al. (Tian et al., 2009) presented malware classification research using classification methods based on sequence information. Sequences were acquired from 1367 samples, including unpackaged trojans, viruses, and clean files. Several classification techniques, including tree-based classifiers, KNN, statistical algorithms, and AdaBoost were used to analyze the information identifying the sequences presented in each sample. Using k-fold cross validation on unpackaged malicious and benign files, the RF classifier achieved an accuracy rate of 97%.

Santos et al. (Santos et al., 2013) offered a new hybrid malware detector that merges the frequency of occurrence of operational codes (statically collected) with information about an executable's execution track (dynamically obtained). This hybrid strategy has been found to increase the performance of both methods when performed individually. KNN, DT, RF, SVM (Support Vector Machine), and NB methods were used in this work. Static, dynamic, and hybrid techniques were evaluated separately. The SVM (Normalized Polynomial Kernel) achieved the highest accuracy for each approximation (Dynamic: %77.26, Hybrid: 96.60%, and Static: 95.90%).

Patil et al. (Patil and Deng, 2020) used a method to extract various feature sets from malware data such as system calls, opcodes, and bytecodes. Work has been done on Microsoft's malware dataset available on the Kaggle website. The dataset contains 10868 malicious files from nine different malware families. The study examines the effectiveness of machine learning algorithms (i.e., DT, RF, NB, KNN, Support Vector Classifier (SVC), Stochastic Gradient Descent (SGD), Logistic Regression (LR)) and deep learning-based (i.e., Deep Neural Network (DNN)) models. Each of the features in the dataset is examined and the findings show that the feature vector for system calls achieves the highest accuracy.

Azeez et al. (Azeez et al., 2021) proposed a method based on collective learning. After applying the CNN classifier, various machine learning algorithms were applied for the final stage. For comparison, RF, NB, DT, GB and Adaboost algorithms were used. The RF algorithm yielded the highest result with an accuracy rate of 99.24%.

A literature review of existing malware detection classification researchs utilizing machine learning techniques was published by Harshalatha et al. (Harshalatha and Mohanasundaram, 2020).

Random Forest (RF), Support Vector Machine (SVM), BayesNet and Multi-Layer Perceptron (MLP) are the classifier models utilized in this study. First, all malware samples were put through a 10 fold cross validation test, and the findings of the classifier showed that RF performed the best with 98% accuracy. It was shown that using the RF classifier for various datasets caused the same RF accuracy to drop by 12%.

Tahtacı and Canbay (Tahtacı and Canbay, 2020) investigated Android-based files with a machine learning model along with N-Gram features. The models were combined with methods to determine the threshold of variance and obtain features information. With the APKTool tool, 3000 Android Package (APK) files were converted to source code and the opcodes of the programs were obtained. DT, KNN, NB, LR, RF and SVM machine learning algorithms are trained using N-Gram features. The best test result is obtained after reducing the number of features in 3-gram to two (100% test score was obtained in the KNN and SVM models).

Table 1. Summary of the related studies on malware detection and their results.

Author	Year	Classifiers	Best Classifier	Accuracy
Tian et al.	2009	RF, NB, DT, IB1 and SVM	RF	0.97
Santos et al.	2013	KNN, DT, RF, SVM and NB	SVM	Static:0.9590 Dynamic:0.7726 Hybrid: 0.9660
Patil et al.	2020	DT, SGD, RF, SVC, LR, DNN, NB, KNN	DNN	>0.95
Tahtacı and Canbay	2020	DT, KNN, NB, LR, RF and SVM	KNN and SVM	1.0
Harshalatha et al.	2020	RF, BayesNet, MLP ve SVM	RF	0.98
Azeez et al.	2021	CNN, RF, NB, DT, GB and Adaboost	RF	0.99

The article continues as follows. The second chapter highlights technique, which includes machine learning-based malware classification and detection methods and dataset preprocessing. The third chapter analyzes machine learning malware detection algorithms and evaluates the findings. This topic is concluded in Chapter 4 by addressing the research aspects.

2. METHODS

The main purpose of this study is to detect malicious software with a data set created by static analysis method. This section contains information about the dataset and machine learning algorithms used for malware analysis.

2.1. Dataset and Data Preprocessing

The data set was obtained from C-Prot Turkey. It consists of 1000 pieces of software labeled as malicious (malware) and 1000 pieces of software labeled as benign (non-malware). In order to extract the data set features used in the study, the features from the files of programs were obtained by applying the static analysis method. The sample fragment of the dataset is shown in table 1. The file size, digital signature status, libraries and functions used in the software were used as dataset features of each software. In this study, the data set was divided into two sub-datasets and 80% (1600 pieces) of the data were used for training and 20% (400 pieces) for testing.

Table 1. A sample dataset before the dataset preprocessing step.

SIZE	DIGITAL SIGNATURE	LIBRARIES	FUNCTIONS	LABEL
323800	1	KERNEL32.dll,USER32.d...	GetSystemTime,PostMessage,FreeLib…	1
6595084	0	MSVCR100.dll,IMM32.dl...	MD5,SHA,SHA1,HMAC,GetSe...	1
139084	0	lib1.dll,bcrypt.dll,KERNEL3...	BCryptOpenAlgorithmProvider,BC...	0
119160	0	kernel32.dll,user32.dll	ExitProcess,VirtualAlloc,VirtualFree…	1
560906	0	uxtheme.dll,gdi32.dll,kern...	CreateWindow,CreateWindowEx,Ge…	1
162733	0	zlib1.dll,bcrypt.dll,KERN…	BCryptOpenAlgorithmProvider,BCryp…	0
511552	1	VCRUNTIME140.dll,KER…	SHA,InitializeConditionVariable…	0

Among the features in the dataset, each of the functions and libraries in the functions and libraries (features) column has been made a separate feature, so that each function and library name is also considered a dataset feature. As a result of this process, 195 unique libraries and 4,327 unique functions were obtained. Including size, digital signature, label, and these unique functions and libraries were also used as features in the dataset. Thus, a total of 4,525 features (4,525 columns) were obtained. Table 2 shows sample dataset after the dataset preprocessing step.

The data preprocessing step transforms the data into a format that machine learning algorithms can analyze efficiently (Markel et al., 2014). In order for the models to work efficiently in machine learning algorithms, categorical inputs need to be converted into numerical expressions. For this reason, normalizing all the features in the dataset to binary values as "0" or "1" provides an advantage in the training process of the model. In this study values of 1 and 0 (value of 1 presence of the feature in the software, 0 value of the absence of the feature in the software) were assigned for each feature according to the content of the software.

Table 2. A sample dataset after the dataset preprocessing step.

SIZE	DIGITAL SIGNATURE	.ctor	KERNEL32.dll	zlib1.dll	GetSystemTime	ExitProcess	LABEL
323800	1	0	1	0	0	0	1
6595084	0	1	0	0	0	0	1
139084	0	0	0	0	1	0	0
119160	0	0	0	0	0	1	1
560906	0	0	0	0	0	0	1
162733	0	0	1	0	0	0	0
511552	1	0	1	1	0	0	0

2.2. Machine Learning Algorithms

With the proliferation of new and unseen malware families, there is a need to develop new methods to detect malware. Machine learning is one of these methods. In this study, well-known machine learning algorithms such as KNN, RF, DT and Gaussian NB were trained and tested to detect malware. The theoretical details of the machine learning methods used in this study are given below.

2.2.1. Gaussian naive bayes

The NB classifier is a Bayesian-based probabilistic classification mechanism. The main goal of classification is to find the best match between a set of new data and a set of classifications within a specific problem domain (Yang, 2018). The relationship in is stated by Bayes' theorem with respect to the class variables f_1 and f_n and the dependent feature vector (1).

C represents the given target, and f represents the features.

$$p(C|\boldsymbol{f_1}, \dots \boldsymbol{f_n}) = \frac{P(C)p(\boldsymbol{f_1},\dots \boldsymbol{f_n}|C)}{p(\boldsymbol{f_1},\dots \boldsymbol{f_n})} \qquad (1)$$

In simple Bayesian classification, the Gaussian distribution is a method of using continuous features. If the feature has continuous values, they are from a Gaussian or normal distribution.

$$\boldsymbol{p}(\boldsymbol{x_i} \mid \boldsymbol{y}) = \frac{\mathbf{1}}{\sqrt{\mathbf{2}\pi\boldsymbol{\sigma_y^2}}}\ \boldsymbol{exp}\left(-\frac{(\boldsymbol{x_i}-\boldsymbol{\mu_y})^2}{\mathbf{2}\boldsymbol{\sigma_y^2}}\right) \qquad (2)$$

Parameters σ(y) and μ(y) shown in formula (2) are estimated using maximum probability.

2.2.2. K-nearest neighbors

Based on the supervised learning technique, KNN is one of the Machine Learning algorithms used for both classification and regression. Based on the similarity between the new state data and the existing states, it classifies the new state into the category to most similar to the existing classes. It computes the distance between the test data and all training points and attempts to predict which category the test data belongs to.

Algorithm selects the number of K points close to the test data. 'K' value computes the probability of test data belonging to training data classes, and then the class with the highest probability is chosen.

To find the nearest neighbors, various distance measurement methods are used (Chumachenko, 2017). The Hamming, Manhattan, Minkowski, and Euclidean distances are all popular. Minkowski distance was used as a distance measure in this study. It then discards the point from among the k nearest neighbors to the class (where k is an integer).

Where x=$(x_1, x_2, x_3, \ldots x_n)$ and y=$(y_1, y_2, y_3, \ldots y)$ and p is an integer.

$$\boldsymbol{D}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} (p_i - q_i)^{\frac{1}{p}} \quad (3)$$

2.2.3. Decision tree

A DT is a recursively expressed machine learning classifier. It consists of nodes that form a root tree (Rokach, Maimon, 2005). The root and internal nodes, branches, and leaf nodes of decision trees have a hierarchical tree structure. It starts with a node and grows a tree structure by adding branches when new results are obtained. The result is achieved by traversing the nodes using the entered value. Figure 1 depicts an example DT.

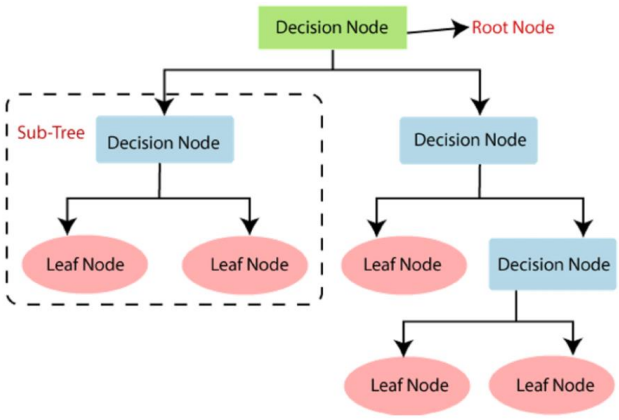


Figure 1. Representation of DT algorithm (Deshpande, 2021).

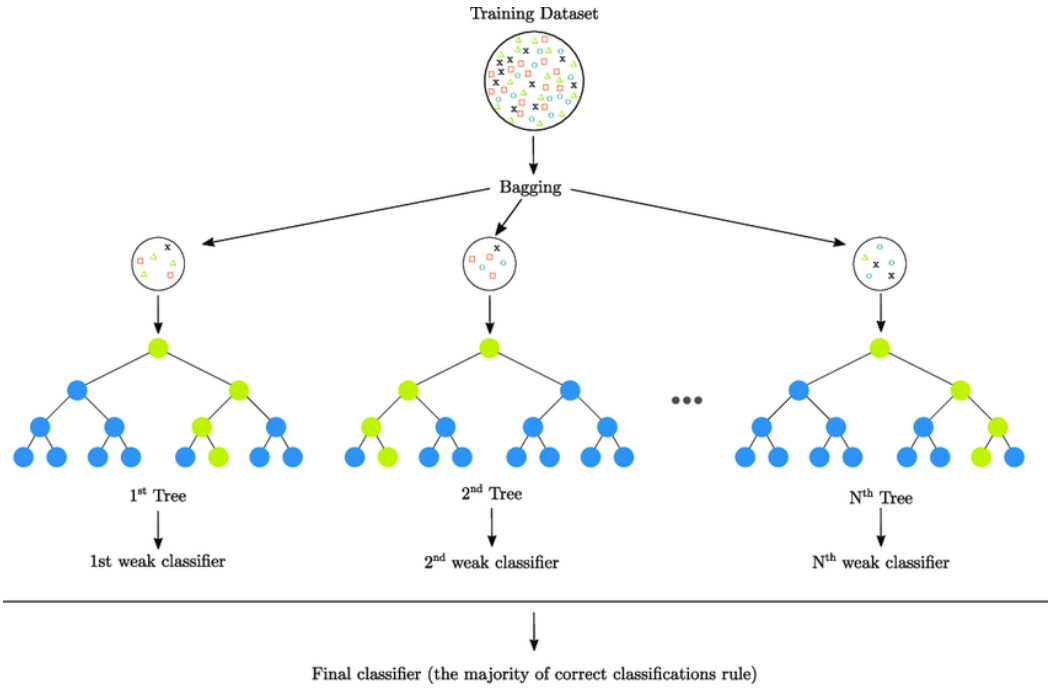


Figure 2. Representation of RF algorithm (Deshpande, 2021).

2.2.4. Random forest

Random forests are a combination of tree estimators that contain a set of DTs in different subsets of a given dataset, where each tree is sampled independently and depends on the values of a random vector with the same distribution for all trees in the forest (Breiman, 2001).

It does not depend on the output of a single decision tree, but aggregates the predictions generated from each tree and predicts the final result based on the majority of these predictions. Increasing the number of trees in the forest prevents overfitting and improves accuracy. Figure 2 presents the RF algorithm.

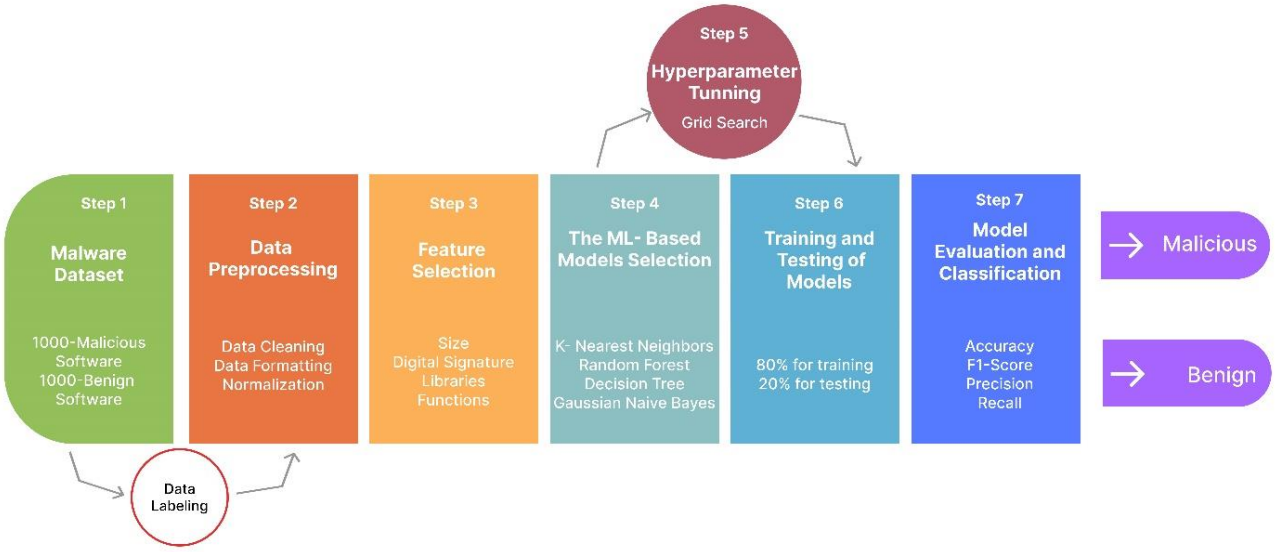


Figure 3. The overall design of malware analysis system used in the study.

3. FINDINGS

All classifiers used in the study were developed and tested with the python-based Scikit learn library (URL-1). The results of the models, trained with KNN, DT, RF and Gaussian NB algorithms, were evaluated with accuracy, f1-score, recall and precision metrics.

Grid search method was applied for hyperparameter optimization during the training of classifiers. The hyperparameters to be tested in the model and the determined value ranges are shown in Table 3. In the grid search method, a model is built with all combinations separately and the model is trained for determining the most successful hyperparameter set according to the specified metric. Grid search hyperparameter optimization technique has been observed to increase model performance. The overall design of malware analysis system used in the study is shown in figure 3.

According to the results; RF (97% - Highest) and DT (94%) algorithms achieved the highest accuracy rate, while NB (53% - Lowest) and KNN (59%) algorithms achieved lower results. RF and DT are more effective than KNN and Gaussian NB for a variety of reasons because they are tree-based algorithms. One argument for this is that the tree structure they produce tends to reduce the inaccuracy rate relative to the previous tree. It is thought that the NB algorithm gives weaker classification results compared to other classification algorithms due to the independent determination of the class values of the features. Figure 3 shows the comparison graph of accuracy, f1-score, precision and recall values of algorithms.

RF and DT algorithms gave the highest accuracy results. There are many reasons that RF algorithm gave higher accuracy than DT. One of the important reason is that random forest algorithm finds the root node and assigns the nodes randomly. Rather than feeding a decision tree with all the data, it can generate a random forest by feeding it piecemeal and multiple DTs.

From Table 5, it is seen that the f1-score, precision and recall values are in similar proportions with the accuracy values of the algorithms. In addition, the high precision and recall values of the DT and RF algorithms prove that they can accurately distinguish between malicious and benign software

Table 3. The hyperparameter values for machine learning algorithms using grid search method.

	Hyperparameter	**Value Range**	**Description**
KNN	N_neighbours Distance Metric	(1,50) Minkowski	Number of neighbors Metric to use for distance computation.
RF	Max_depth Max_features N_estimators **criterion**	(1,10) [3, 5, 10, 15] [100, 200, 500, 1000, 2000] gini	The maximum depth of the tree. The number of features when looking for the best split. The number of trees in the forest. The function to measure the quality of a split.
DT	Max_depth	(1,10)	The tree's maximum depth.

	Min_samples_split	(2,50)	The minimum number of samples required to split an internal node
	criterion	gini	The function to measure the quality of a split.
NB	**priors**	**default**=None	Prior probabilities of the classes.
	var_smoothing	**default**=1e-9	Portion of the largest variance of all features that is added to variances for calculation stability

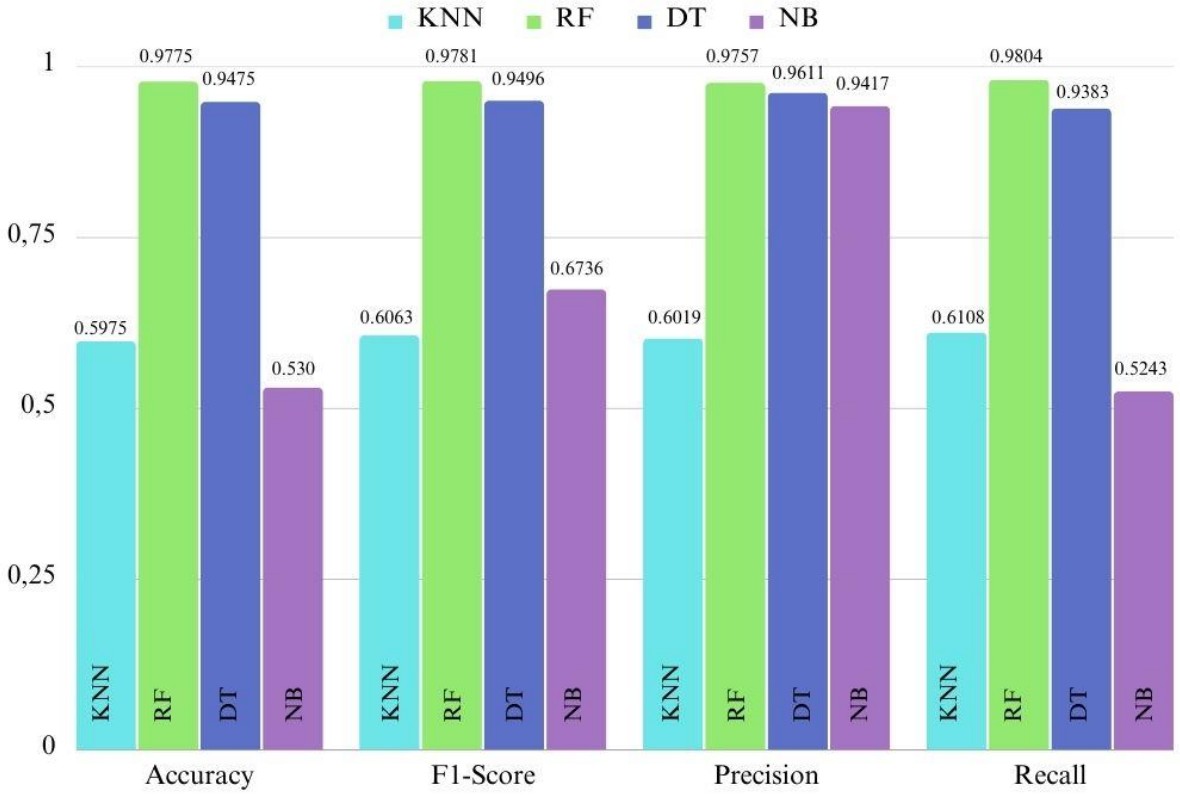


Figure 3. The comparison graph of accuracy, f1-score, precision and recall values of algorithms.

Table 5. Accuracy, f1-score, precision and recall values of algorithm.

Algorithm	Accuracy	F1-Score	Precision	Recall
KNN	0.5975	0.6063	0.6019	0.6108
RF	**0.9775**	**0.9781**	**0.9757**	**0.9804**
DT	0.9475	0.9496	0.9611	0.9383
NB	0.530	0.6736	0.9417	0.5243

The difference between the macro and micro average performance measures is that the macro average weights each class equally, while the micro average weights each feature equally. When Figure 5 and Table 5 are examined, it is seen that the balanced sample numbers of the classes in the data set cause the macro and micro averages to result in the same or very close results.

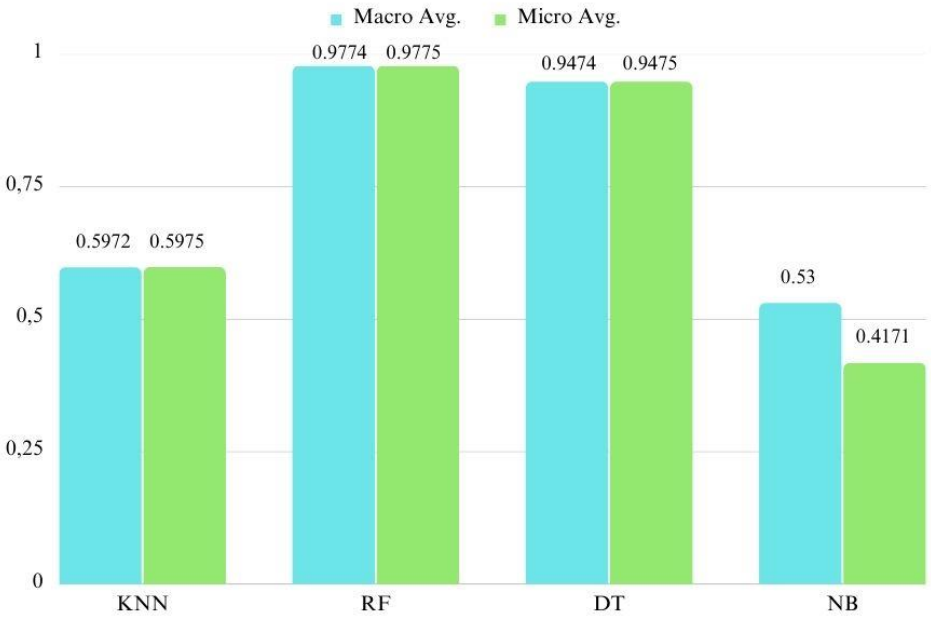


Figure 5. The comparison graph of macro average and micro average values of algorithms.

The RF algorithm gave the best results with 97.75% micro average and 97.74% macro average value.

Table 5. Macro average and micro average values of algorithm.

Algorithm	Macro Average	Micro Average
KNN	0.5972	0.5975
RF	**0.9774**	**0.9775**
DT	0.9474	0.9475
NB	0.53	0.4171

4. RESULTS

The main purpose of this paper is to detect malicious software using static analysis and machine learning methods (i.e., DT, KNN, RF, NB). The data obtained by static methods were processed and converted into a suitable format for training with machine learning algorithms, and then the models were trained with this data set and analyzes were carried out. The experimental results of the trained and tested machine learning algorithms are compared. It has been seen that tree-based DT and RF algorithms show high performance in malware analysis.

In future work, the dataset can be trained by developing deep learning-based models to improve accuracy, f1-score, precision and recall metrics and higher malware detection rates.

APPENDIX

We would like to thank C-Prot Turkey Company for sharing the dataset used in this study.

REFERENCES

Azeez, N. A., Odufuwa, O. E., Misra, S., Oluranti, J., & Damaševičius, R. (2021). Windows PE malware detection using ensemble learning. In *Informatics* (Vol. 8, No. 1, p. 10). MDPI.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5-32.

Chumachenko, K. (2017). Machine learning methods for malware detection and classification.

Deshpande, N. M., Gite, S., & Aluvalu, R. (2021). A review of microscopic analysis of blood cells for disease detection with AI perspective. *PeerJ Computer Science*, *7*, e460.

Gandotra, E., Bansal, D., & Sofat, S. (2014). Malware analysis and classification: A survey. *Journal of Information Security*, *2014*.

Harshalatha, P., & Mohanasundaram, R. (2020). Classification Of Malware Detection Using Machine Learn-ing Algorithms: A Survey. *International Journal of Scientific & Technology Research*, *9*(02).

Hassen, M., Carvalho, M. M., & Chan, P. K. (2017, November). Malware classification using static analysis based features. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)* (pp. 1-7). IEEE.

Maimon, O., & Rokach, L. (Eds.). (2005). Data mining and knowledge discovery handbook.

Markel, Z., & Bilzor, M. (2014, October). Building a machine learning classifier for malware detection. In *2014 second workshop on anti-malware testing research (WATeR)* (pp. 1-4). IEEE.

Patil, R., & Deng, W. (2020, March). Malware analysis using machine learning and deep learning techniques. In *2020 SoutheastCon* (Vol. 2, pp. 1-7). IEEE.

Santos, I., Devesa, J., Brezo, F., Nieves, J., & Bringas, P. G. (2013). Opem: A static-dynamic approach for machine-learning-based malware detection. In *International joint conference CISIS'12-ICEUTE´ 12-SOCO´ 12 special sessions* (pp. 271-280). Springer Berlin Heidelberg.

Sapountzoglou, N., Lago, J., & Raison, B. (2020). Fault diagnosis in low voltage smart distribution grids using gradient boosting trees. *Electric Power Systems Research*, *182*, 106254.

TAHTACI, B., & CANBAY, B. (2020, October). Android malware detection using machine learning. In *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)* (pp. 1-6). IEEE.

Tian, R., Batten, L., Islam, R., & Versteeg, S. (2009, October). An automated classification system based on the strings of trojan and virus families. In *2009 4th International conference on malicious and unwanted software (MALWARE)* (pp. 23-30). IEEE.

Yang, F. J. (2018, December). An implementation of naive bayes classifier. In *2018 International conference on computational science and computational intelligence (CSCI)* (pp. 301-306). IEEE.

URL-1: https://scikit-learn.org/stable/
[Erişim Tarihi: 15.05.2023]

BİLGİSAYAR BİLİMLERİ VE TEKNOLOJİLERİ DERGİSİ
JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGIES
http://dergipark.org.tr/tr/pub/bibted

Araştırma Makalesi

Örtülü Yüzlerin Tanınmasında Haar Cascade ve MongoDB Entegrasyonuyla Geliştirilen Yüz Tanıma Sisteminin (YTS) Performans Değerlendirmesi

Anıl YILDIZ[1], Zafer GÜNEY[2], Hakan AYDIN*[2]

[1]İstanbul Gedik University, Department of Information Security Technology, Istanbul, 34020, Türkiye
[2]İstanbul Topkapı University, Department of Computer Engineering, Istanbul, 34020, Türkiye

Anahtar Kelimeler:
Yüz tanıma teknolojisi
Maske takma
COVID-19 pandemisi
Haar Cascade yöntemi
MongoDB veritabanı

ÖZ

Her ne kadar geleneksel yüz tanıma sistemleri (YTS) maske takılıp takılmadığını belirli bir başarı oranında belirleyebilseler de maske takan kişilerin yüzlerinin büyük bir kısmının örtülü olması nedeniyle başarısız olabilmektedirler. Özellikle maske takan bireylerin yüzlerinin önemli bir kısmının örtülü olmasından kaynaklanan zorluklar mevcut YTS'lerin performansını sınırlamaktadır. Bu araştırmada maske takılmış yüz tanıma için Haar Cascade yönteminin OpenCV kütüphanesi kullanılarak gerçek zamanlı olarak MongoDB veritabanı ile entegrasyonun yapılması ve bu durumun kapsamlı deneylerle performansının ortaya konması amaçlanmıştır. Deneylerde maskeli yüzlerin büyük bir kısmının örtülü olduğu gerçekçi yüz görüntülerinden bu çalışma kapsamında oluşturulan veri seti kullanılmıştır. Araştırmamız, yüz tanıma doğruluğunun maskelenmiş yüzler için %85, maskesiz yüzler için %61 ve yüzün yarısı farklı bir nesne tarafından kapatıldığında %41 olduğunu göstermiştir. Bu çalışmanın Haar Cascade yöntemini gerçek zamanlı veritabanı yönetimi entegrasyonuyla birleştirerek daha etkili ve uygulanabilir bir maske tespit çözümü sunması bağlamında literatüre katkı sağlayacağı değerlendirilmektedir.

Performance Evaluation of Face Recognition System (FRS) Developed with Haar Cascade and MongoDB Integration in Recognition of Covered Faces

Keywords:
Face recognition
Mask wearing
COVID-19 pandemic
Haar Cascade method
MongoDB database

ABSTRACT

Although traditional face recognition systems (FRS) can detect with a certain success rate whether a mask is worn, they may fail because most of the faces of the people who wear masks are covered. The difficulties arising from the fact that a significant part of the faces of individuals wearing masks are covered limits the performance of existing FRSs. This research, it is aimed to integrate the Haar Cascade method with the MongoDB database in real-time using the OpenCV library for mask-wearing face recognition and to demonstrate its performance with extensive experiments. In the experiments, the data set created within the scope of this study from realistic face images, in which most of the masked faces are covered, was used. Our research has shown that the accuracy of face recognition is 85% for masked faces, 61% for unmasked faces, and 41% when half of the face is covered by a different object. It is considered that this study will contribute to the literature in terms of providing a more effective and applicable mask detection solution by combining the Haar Cascade method with real-time database management integration.

***Sorumlu Yazar**

(anil.yildiz@gedik.edu.tr) ORCID 0000-0003-4607-6660
(zaferguney@topkapi.edu.tr) ORCID 0000-0003-1974-4264
*(hakanaydin@topkapi.edu.tr) ORCID 0000-0002-0122-8512

e-ISSN: 2717-8579
Geliş Tarihi: 08/08/2023; Kabul Tarihi: 22/09/2023

1. INTRODUCTION

It can be observed that the COVID-19 pandemic, which is a significant health crisis affecting people worldwide, is still ongoing in different regions and different ways today. This outbreak has been experienced and learned as a health crisis that affects the health of all people worldwide. COVID-19 is a respiratory illness caused by a coronavirus called SARS-CoV-2. It is a disease in which the virus spreads through respiratory droplets when an infected person coughs, sneezes, or talks. Another important aspect that has been experienced is that the course of this outbreak can vary depending on the measures taken by countries. One of these measures is the wearing of masks by people, which has been experienced and learned as another important aspect. Wearing masks is essentially an effective measure against COVID-19 and has emerged as another significant issue during this pandemic. The World Health Organization (WHO) and other health authorities emphasize that wearing masks plays an important role in reducing the risk of infection. Wearing a mask not only reduces the risk of transmitting the disease by preventing the spread of respiratory droplets in the air but also reduces the risk of inhaling respiratory droplets from others and getting infected. The importance of wearing masks is particularly emphasized in indoor or crowded environments. Masks should be used to protect both yourself and others around you. Furthermore, wearing masks, along with other measures such as hand hygiene and social distancing, can further reduce the risk of infection. Measures to be taken against COVID-19 include social distancing, mask usage, hygiene practices, limitation of mass gatherings, testing and monitoring measures, and travel restrictions (Ciotti et al., 2020). It has been observed that wearing masks is an important strategy in controlling the spread of COVID-19 during the pandemic and that mask usage reduces the risk of infection and transmission by reducing the spread of respiratory droplets. Additionally, during this global pandemic, it has been learned through experience that mask types, proper mask usage, and wearing masks are crucial in the fight against this disease. Particularly, it has been experienced and learned that mask usage is important in public areas and situations involving contact with infected individuals, and it contributes significantly to controlling the outbreak. The detection of whether masks are worn or not is of great importance in controlling the outbreak and ensuring safety in public areas.

Face recognition systems (FRS) are technologies that perform authentication or recognition by analyzing the characteristics of people's faces. Various techniques are currently being developed, including local, holistic, and hybrid approaches, which define a face image using only a few or all of the facial image features (Kortil et al., 2020). Face recognition involves detecting faces in an image or video and associating these faces with specific people (Parmar et al., 2014). These systems typically use an image taken from a person's face via a camera or image sensor. Face recognition algorithms mathematically analyze the unique features of the face, such as the shape of the face, eye position, nose structure, and lips, and create a unique facial signature for a person. This signature is then compared with other recorded facial images for matching and authentication or recognition is performed. Face recognition involves a series of processes and analyses carried out to detect faces in an image or video, extract unique features of faces, and identify or verify a particular person using these features (Grudin, 2000). Face recognition usually consists of two stages: face detection, which is the process of detecting faces in an image or video, and face recognition, which involves matching and identifying specific people (Parmar et al., 2014). Facial recognition systems are used in various applications such as security, authentication, access control, and crime prevention. Research is currently being conducted on facial recognition technology in connection with the topic of wearing masks. Traditional face recognition systems perform authentication or recognition of individuals by analyzing the characteristic features of their faces. However, since most of the face is covered, the accuracy and effectiveness of these systems may decrease when dealing with people wearing masks. For this reason, it is necessary to develop new approaches to update face recognition systems or detect mask-wearing. In some studies, different algorithms and methods have been used to detect whether a mask is worn. Various studies are being carried out to enable facial recognition systems to produce more accurate results for individuals who currently wear masks and to detect mask-wearing. During the COVID-19 pandemic, facial recognition technology has become an important tool for controlling the spread of infection and maintaining security in public spaces (Van Natta et al., 2020). Facial recognition technology is important during the COVID-19 pandemic because it can detect mask-wearing, helping to control the spread of infection and ensure safety in public areas (Mundial et al., 2020). In this context, the detection of mask use by using face recognition systems has recently become a research and examination subject.

This study aims to integrate the Haar Cascade method with the OpenCV library and integrate it with the MongoDB database in real-time for accurate recognition of masked faces, and to evaluate the performance of this approach with extensive experiments and sample application scenarios. Python, MongoDB, and OpenCV, which are the subject of this study, are popular technologies used in different but compatible areas. Python is a user-friendly and easy-to-understand programming language that is widely used to develop various applications thanks to its extensive library support. OpenCV is a library for image and video processing

applications. It integrates with Python and supports a wide variety of visual analytics such as image processing, object detection, face recognition, and video analytics. Thus, it provides powerful tools for manipulating, analyzing, and manipulating image and video data. Experiments carried out within the scope of the study were carried out using a dataset created from realistic facial images, taking into account different lighting conditions. This dataset aims to more robustly evaluate the effectiveness of the proposed method by including cases where most of the masked faces are covered. MongoDB is a document-based NoSQL database that provides a scalable, flexible, and high-performance data storage solution. It works with unstructured data models and is ideal for processing large datasets. This database is also equipped with features such as high-performance query processing, backup, and replication. The obtained results confirm that the proposed method can be used successfully in real-time applications and its effectiveness in accurately recognizing masked faces. Although our research has decreased, it offers an innovative solution for the detection of mask-wearing, which continues to be an important need within the scope of the COVID-19 process, which still has an impact in many parts of the world. The method proposed in our study can be used in areas such as security, health, and public space management and contributes to the development of relevant research and applications in these areas. In addition, it is considered that the applied approach presented in our study on the compatible use of Python and MongoDB will be a reference for researchers and developers using these technologies.

2. METHOD

This study addresses the existing limitations in recognizing obscured faces, particularly focusing on the challenges posed by a significant portion of the face being concealed due to mask-wearing. The aim is to overcome these challenges and develop a system capable of effectively recognizing masked faces. To achieve this goal, the Haar Cascade method was implemented using the OpenCV library, and simultaneous integration of recognized facial data into a MongoDB database was accomplished. During the data collection phase, a dataset representative of real-world scenarios involving diverse degrees of face masking was curated. The Haar Cascade method was selected as the preferred algorithm for facial detection and extensively evaluated using the compiled dataset. During this phase, performance metrics such as facial detection accuracy, false positives/negatives, and processing times were meticulously analyzed. Additionally, through the integration with the MongoDB database, the real-time storage and management of recognized facial data were realized. Performance evaluation was conducted rigorously on the curated dataset, measuring the recognition efficacy of the Haar Cascade method against various performance criteria. These criteria encompassed accuracy rates, erroneous positive and negative outcomes, and processing speeds.

The work's summary is depicted in **Figure 1**, which presents the main steps. First, camera images are captured in real time. Then, facial recognition is carried out using Haar Cascade and OpenCV. The goal is to accurately identify faces, especially those wearing masks. Next, recognized face data is stored in a MongoDB database for future access. Lastly, the system's performance is evaluated through experiments to analyze accuracy, speed, and reliability impacts.

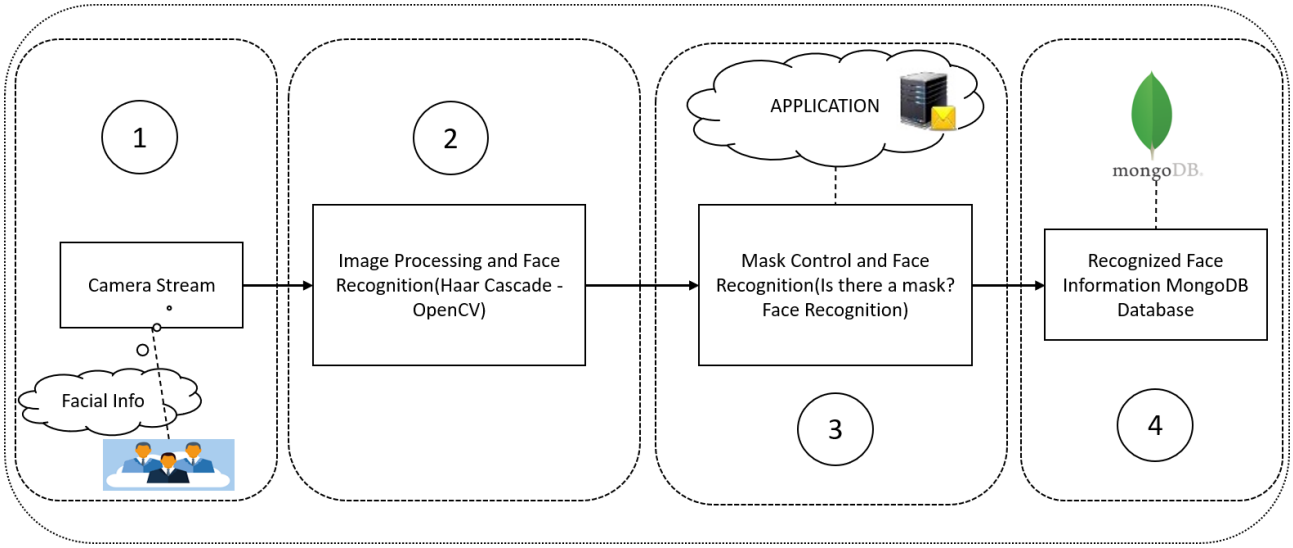


Figure 1. The block diagram summarizing the study

The system's performance was validated through illustrative real-world applications, effectively simulating real-time conditions. The results obtained from experiments were presented with thorough analysis and comparative discussions. The contributions of this study to the scientific discourse and its potential applications were emphasized. In particular, the amalgamation of the Haar Cascade method with MongoDB integration was deemed to hold the potential to yield an enhanced solution for effective mask detection, offering an avenue for practical applications.

3. RELATED WORKS

In the study conducted by Mundial et al. (2020), the facial recognition problem during the COVID-19 pandemic was addressed. The authors presented information about the challenges faced by facial recognition technologies in accurately identifying individuals wearing masks. The article discusses different approaches, methods, and techniques for recognizing individuals wearing masks and focuses on the impact of mask-wearing on facial recognition systems. Additionally, the article addresses the security, privacy, and ethical issues that have emerged with the increased use of facial recognition technologies during the pandemic. It provides insights into the current challenges and future potential of facial recognition technologies in the context of the COVID-19 period. In the study conducted by Karadağ (2020), face recognition was performed by detecting the face with the Haar-Cascades classifier and then using the LBPH (Local Binary Patterns Histograms) method. Van Natta et al. (2020) examined the rise and regulation of thermal facial recognition technology during the COVID-19 pandemic. The authors studied the increasing use of thermal facial recognition technology, its applications in security and healthcare sectors, ethical and privacy concerns, and regulatory measures. They also discussed the potential benefits and risks of facial recognition technology. Vansh et al. (2020) discussed a face recognition method developed using the "YCbCr" and "Adaboost" algorithms. The article states that the use of the YCbCr color space enhances face detection performance, and the AdaBoost algorithm improves classification accuracy. This study presents a new approach to enhance the performance of face recognition systems and achieve more accurate and efficient results. In the study conducted by Özdemir and Koç (2019), emotion recognition was performed using deep learning methods for seven different facial expressions with a dataset (RidNet) created using publicly accessible images from the internet. Then, transfer learning was performed over RidNet with well-known convolutional neural network architectures in the literature such as AlexNet, GoogLeNet, and ResNet101. The Compound Facial Expressions of Emotion (CE) and Static Facial Expressions in the Wild (SFEW) datasets were designated as test datasets. Firstly, the convolutional neural network architecture, which shows the best classification performance, was determined by experimental studies. This convolutional neural network is trained with AffectNet, The Karolinska Directed Emotional Faces (KDEF), and RidNet. Similar classification performances were obtained when networks trained with AffectNet, KDEF, and RidNet were tested with a dataset (CE) created in a controlled environment. In the test dataset (SFEW) in the uncontrolled environment, the network trained with RidNet provided a significant advantage over other networks. Kekül et al. (2018) compared Eigenfaces and Artificial Neural Networks in face recognition applications. The article highlights the effectiveness and different advantages of both methods in face recognition performance. The Eigenfaces method stands out for its fast operation and low computational requirements, while artificial neural networks are capable of detecting more complex features and providing higher recognition accuracy. The article provides a detailed comparison of these methods in terms of their applications, advantages, and disadvantages. It emphasizes that this study provides a guiding framework for the decision-making process in face recognition systems and suggests that better results can be achieved by employing different methods. Mamak et al. (2020) researched a real-time face recognition-based personnel control and monitoring system. The article explains how facial recognition technology can be used in personnel control and monitoring processes and describes the functioning of the designed system. The designed system is used for real-time facial recognition in applications such as personnel records, attendance tracking, and access control. The article also provides information about the techniques and algorithms used in the system design. Atasoy and Tabak (2018) developed a face recognition application using Support Vector Machines (SVM). The article extensively discusses how the SVM algorithm can be applied in the field of face recognition, including its working principles and application stages. The process of data collection for face recognition, feature extraction, training, and classification steps are emphasized. The article provides information about the performance, accuracy rate, and advantages of SVM in face recognition, along with sharing the results of experiments conducted on different datasets. In the study conducted by Terzopoulos and Waters (1990), the analysis of facial images using physical and anatomical models was carried out. The article focuses on how facial recognition systems can be based on physical and anatomical models to obtain more accurate and reliable results. It is stated that physical models represent the structure and movement of the face with mathematical equations and are used in the analysis of facial movements. Anatomical models, on the other hand, include databases representing the anatomical structures of the face and feature-based analysis methods. The

article presents research and results on how these models can be used to improve facial recognition performance and their effects on the analysis process. In the study conducted by Kirby and Sirovich (1990), a method was developed using the Karhunen-Loeve procedure for characterizing human faces. Rodriguez and Marcel (2006) performed a study on face verification using aligned local binary pattern histograms. In the study conducted by Ngo et al. (2009), a modular approach was used to implement an FPGA-based design for a real-time face detection system. These studies, examined within the scope of our research, indicate that during the COVID-19 pandemic, research was conducted on the challenges of accurately recognizing individuals wearing masks using facial recognition technology and addressing the security, privacy, and ethical issues of this technology. It is also understood that studies were conducted on enhancing the performance of facial recognition systems through the use of different methods and developing FPGA-based designs for real-time applications.

In our research, studies on the Python programming language and MongoDB database were also examined. In the study conducted by Araujo et al. (2021), an analysis comparing the performance of NoSQL Cassandra and MongoDB databases was conducted. The book "Programming Python" by Lutz (2010) focuses on the Python programming language and object-oriented programming. The study conducted by Lemenkova (2020) discusses the use of Python libraries such as matplotlib, seaborn, and pandas for visualizing geographic datasets generated by QGIS. In Taylan's (2014) article, it is stated that Python is the most commonly taught programming language in universities in the United States. Kolakowski (2020) provides information about the popularity of Python among the popular programming languages on GitHub and its ranking on the platform. Saran and Saran (2019) conducted a comparative security analysis of NoSQL databases such as Cassandra and MongoDB. In the study by Daşdemir and Kara (2019), the performance of MongoDB, a NoSQL database, was analyzed under different workloads. The study presents information on evaluating MongoDB's performance based on various metrics and criteria, and how it behaves under different workloads. The book "MongoDB" defines MongoDB as a document-oriented NoSQL database. The book extensively covers MongoDB's features, data modeling, query language (MongoDB Query Language - MQL), and other related topics (Mongo, 2015). The book "MongoDB in Action: Covers MongoDB Version 3.0" also defines MongoDB as a document-oriented NoSQL database. The book highlights MongoDB's flexible data model, high-performance query capabilities, horizontal scalability features, and other advantages (Banker et al., 2016). In the study "MongoDB vs Oracle--database comparison," MongoDB is also defined as a document-oriented NoSQL database. The study evaluates MongoDB's flexible data model, high performance, scalability, and ease of data management by comparing it with the Oracle database (Boicea et al., 2012). The book "MongoDB Data Modeling" published by França (2015) focuses on the topic of data modeling in MongoDB. The book explains the concept of NoSQL databases and specifically discusses data modeling strategies for MongoDB. It explores how the data structure in document-oriented databases should be designed, different approaches from relational databases, and best practices. Additionally, the book provides detailed explanations of MongoDB's data modeling tools and techniques. It emphasizes that MongoDB is a document-oriented NoSQL database and can be used for data modeling strategies. In the studies examined within the scope of our research, the studies highlight the need to consider security, privacy, and ethical issues with the increased use of facial recognition technology during the pandemic. The examined studies reveal that research has been conducted on developing new algorithms and methods to improve the performance of facial recognition systems, the widespread use of thermal facial recognition technology, and the utilization of different databases and feature-based analysis methods.

4. THE PROPOSED SYSTEM

This section outlines the new system we have developed to improve the recognition of faces when partially covered by masks. To achieve this, we combined the Haar Cascade method with the MongoDB integration. The system we propose has two main parts: the part that recognizes faces using the Haar Cascade method, and the part that helps store and manage data using MongoDB. The facial recognition part uses the Haar Cascade method, which is a well-known way of identifying different parts of the face such as the eyes and nose. This helps you find faces in real-time, even if they are partially covered. This part can handle different lighting conditions and how much of the face is covered, making it good for real situations. The MongoDB part allows us to store and manage the face data we find. This includes recording pictures of faces we recognize and details such as when we saw them and how confident we were that they were correct. This way, we can keep everything up to date in real-time and make sure that the data is consistent and ready for future use. The system works step by step to achieve its goals. Initially, it takes live images or video feeds where faces can be hidden by masks. Using the Haar Cascade method, the system identifies faces in these images by precisely positioning facial features, even if they are partially hidden. Recognized faces are then compared to known profiles. Once a match is identified, the system seamlessly integrates the recognized faces and related data into the MongoDB database in real-

time. This database integration is crucial in maintaining data accuracy and consistency. The MongoDB component plays a crucial role in effectively organizing and managing face data, enabling fast queries, data analysis, and future enhancements to the face recognition model. This comprehensive process leverages the power of Haar Cascade and MongoDB to improve facial recognition performance while adapting to real-world scenarios, ultimately contributing to a practical and efficient system.

5. EXPERIMENTAL STUDIES

The experiments included various levels of covering of masked faces under different light conditions based on real-world scenarios. The dataset used for the experiments includes masked face images that reflect real-world conditions. These images represent a wide range of occlusion levels and various light conditions. The data set is enriched with the necessary labels to objectively evaluate the detection results on each face image. Experiments were meticulously conducted under different predetermined occlusion levels and various light conditions. Each experiment was performed on a dataset containing a wide range of facial images. Face detection was performed on each image using the Haar Cascade method. Experimental results were rigorously evaluated using comprehensive performance metrics such as detection success for each image, false positive/negative results, and processing times. The detailed results of these experiments helped us to understand how effective the developed face recognition system is in different covering levels and light conditions. The data obtained constitute an important source of information in terms of identifying weak points in the system and identifying areas where future developments can focus.

In our study, firstly, Python was downloaded from the official website and installed on the computer. Then, the OpenCV library was installed for facial recognition and image processing. OpenCV was downloaded and installed from the Python package manager using pip. MongoDB database was used for data storage and management. MongoDB Community Edition was downloaded from the official website and installed. MongoDB, a NoSQL-based database, was preferred for storing and managing the results obtained in the study. The Haar Cascade method is a feature-based classification algorithm used for image processing and object detection. This method utilizes a pre-trained classifier to recognize specific objects. Haar Cascade consists of a step-by-step process involving feature extraction and classification stages. The Haar Cascade method is based on Haar-like features that define different characteristics of an object, such as its horizontal edges, vertical edges, corners, etc., represented by black and white rectangular regions. These features are applied to the image through sliding and scaling operations. Using a pre-trained classifier, training is performed on a dataset containing positive and negative examples. Positive examples represent images of the desired object, while negative examples represent images without the desired object. The training process involves adjusting the classifier to differentiate the desired object based on these features. During the object detection stage, the trained classifier scans the image to detect potential object locations. These detected regions are then scaled to a higher level of detail for further examination. As a result, the identified objects are recognized with a certain level of precision and accuracy. The Haar Cascade method is widely used in various applications, including face recognition, object detection, and even eye tracking. Pre-trained classifiers can often be found in image processing libraries such as OpenCV and can be easily implemented using these libraries. Haar Cascade is known as a fast and effective object detection method for real-time applications.

Table 1 shows the setting parameters used at different stages of the study and their values. In the study, the adjustment parameters and values of the Haar Cascade method used for face detection are given.

Table 1. Setting Parameters and Values

Method and Purpose	Parameters and Values
Face Detection with Haar Cascade	Scale Factor: 1.1 Min Neighbors: 3 Min size: (30, 30)
MongoDB Integration and Database	Link URL: mongodb://localhost:27017 Username: user123 Password: password123
Performance Evaluation	Recognition Accuracy: 85% Speed: 0.25 seconds/frame False Positive Rate: 10% False Negative Rate: 5%

According to the information presented in the table, the Scale Factor parameter determines how much the picture jumps at each scale level and the value used here is 1.1. The Min Neighbors parameter indicates how many neighbors are required for each face region during detection, and its value is 3. The Min Size parameter refers to the minimum size of the face to be detected and is indicated here as (30, 30). Under the heading "mongoDB Integration and Database", the setting parameters and values used in the real-time database integration of the study are presented. The link URL parameter refers to the URL used to connect to the MongoDB server and is given as an example, "mongodb://localhost:27017". The username parameter represents the username used to access the database and is designated "user123". The password parameter contains the password of the specified user name and is shown as "password123". Under the title of "Performance Evaluation", the metrics used to evaluate the

performance of the developed system and the results obtained are explained. The Recognition Accuracy metric reflects the accuracy of recognized faces of the developed system and is shown here as 85%. The speed metric expresses the average time for the recognition process of each image and its value is stated as 0.25 seconds/frame. The False Positive Rate metric reflects the percentage of system false positive results and is given here as 10%. The False Negative Rate metric represents the percentage of faces that should be recognized but not mistakenly recognized, and its value is 5%.

As part of the experimental steps, a dataset consisting of facial photographs of individuals wearing and not wearing masks was prepared for facial recognition study. An example image from the dataset is shown in **Figure 2**.

Figure 2. An example image used from the dataset

Face detection and preprocessing steps were performed using OpenCV. The face detection algorithm of OpenCV was used to detect faces. In this step, faces were cropped and resized, and necessary preprocessing steps were applied. The Haar Cascade method was used in the study. It can detect faces quickly and effectively by using predefined features with a pre-trained classifier. However, it may occasionally produce false positive or false negative results. It may provide less accuracy, especially in cases where faces vary in angles, positions, and sizes. Therefore, this method was preferred in our study. In the face recognition step, pre-trained face recognition models in OpenCV were used. These models include algorithms used to recognize faces. Pre-trained models in OpenCV for face recognition were used by employing the Haar Cascade method. In this context, first, the OpenCV library was included in our project. Then, the Haar Cascade model was loaded for face recognition. OpenCV provides pre-trained Haar Cascade files. The "haarcascade_frontalface_default.xml" file is a commonly used model for face detection. Afterward, images were loaded into the system for face detection. The system status regarding an image of a person without a mask is shown in **Figure 3**.

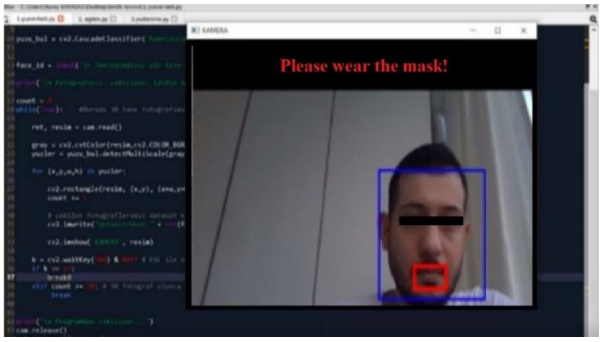


Figure 3. An Unmasked Image

In **Figure 4**, the system status regarding an image of a person wearing a mask is shown

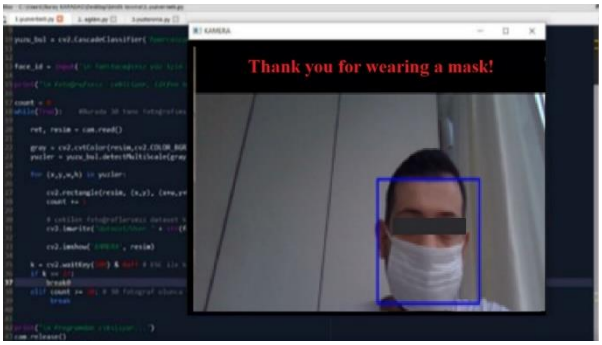


Figure 4. A Masked Image

These pieces of information were stored in the MongoDB database to preserve and manage the results. A suitable MongoDB driver in Python was used to save the results in MongoDB. The outcomes of the experimental studies were documented as records and then stored in the MongoDB database. In the software code utilized for face recognition, the 'detectMultiScale' function was employed to detect faces in grayscale images. The 'scaleFactor' controls the reduction rate of the image scale, while 'minNeighbors' indicates the required number of neighbors for each candidate rectangle. 'minSize' specifies the minimum size of a face. Finally, the results obtained are stored in the database. Each image contains the identity of the detected face, whether a mask is worn or not, and other relevant data. In this way, the necessary steps and experimental processes for the face recognition study conducted within the scope of the research have been completed. The face recognition process was carried out using Python, OpenCV, and MongoDB, and the results were stored in the MongoDB database. The system state of the image of a person without a mask with half of the face covered is shown in **Figure 5**.

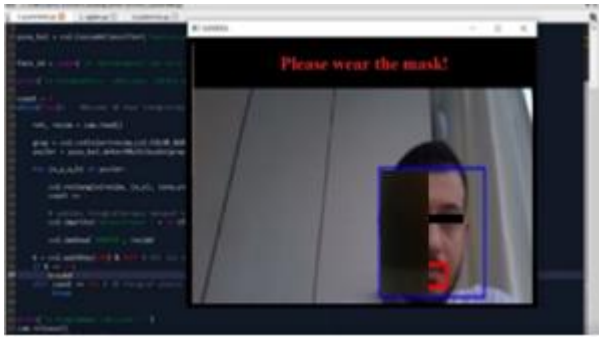


Figure 5. A Half Face Covered Image

6. RESULTS AND DISCUSSION

The operations conducted within the scope of the experimental studies are presented in **Table 2**.

Table 2. Experimental Results

Experiments	Explanation
Experiment 1: Preparation of the dataset.	A dataset consisting of face photographs of individuals wearing and not wearing masks was prepared. Each photograph had a label indicating whether a mask was worn or not.
Experiment 2: Face detection and preprocessing steps.	Face detection and preprocessing steps were performed using OpenCV. The face detection algorithm in OpenCV was used to detect faces.
Experiment 3: Utilization of the Haar Cascade method for mask classification	This method was used to determine whether faces had masks or not.
Experiment-4: Utilization of pre-trained models for face recognition	The pre-trained face recognition models in OpenCV were used, which include algorithms for recognizing faces.
Experiment 5: Creation of the database and assignment of identities	A database was created for the face recognition process, and each person was assigned an identity (ID).
Experiment-6: Determination of identities for detected faces	The detected faces were compared with the faces in the database to determine their identities.
Experiment-7: Saving the results to the MongoDB database	The results were saved to the MongoDB database using a suitable Python driver.

In the first experimental phase, a data set was created for face recognition and mask classification processes. This dataset includes face photographs of individuals wearing and not wearing masks. There are tags on each photograph indicating whether the person concerned is wearing a mask or not. In the second experimental phase, face detection and preprocessing steps were performed. At this stage, face detection was performed using the OpenCV library. OpenCV's powerful face detection algorithm has been used to successfully detect faces in the dataset. In the third experiment, the Haar Cascade method was used to determine whether there were masks on the faces. This method can detect the presence of a mask by detecting different features on the face. In the fourth experiment, face recognition was performed using pre-trained face recognition models in OpenCV. These models include various algorithms for recognizing faces, and the purpose of the experiment is to evaluate the performance of these models. In the fifth step, a database was created for face recognition, and a unique identity (ID) was assigned to each individual. This step is important for associating and tracking recognizable faces. In the sixth experiment, the detected faces were compared with the faces in the database, and their identities were determined. This stage ensures that the accuracy and reliability of the face recognition algorithm are tested. In the seventh experiment, the results obtained were saved in the MongoDB database using a suitable Python driver. This step ensures that the results are stored and made accessible for later analysis.

In the experiments conducted shown in **Figure 6**, the average success rate of "Perception of a Single Person's Face While Masked", "Perception of a Single Person's Face While Unmasked" and "Perception of Half of the Face While Covered with a Different Object" was also obtained.

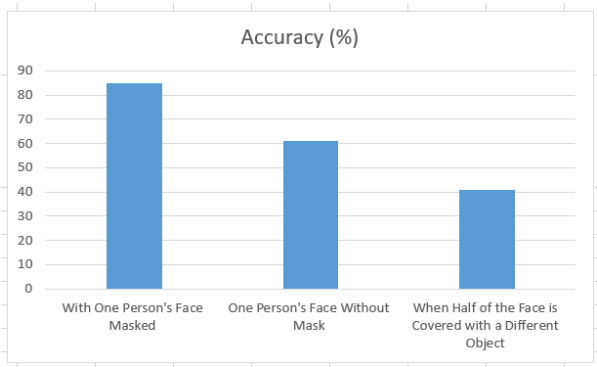


Figure 6. Evaluation of performed experiments

The experiment involving the recognition of a single individual's face while wearing a mask resulted in an accuracy of 85%. This finding signifies the system's capability to effectively identify faces even when a mask is worn, achieving a commendable accuracy level. When assessing unmasked faces, the accuracy remained at 61%, indicating that the recognition of faces without masks also demonstrated a substantial level of success. On a different note, the experiment where half of the face was concealed with a distinct object yielded an accuracy rate of 41%. This outcome implies that recognizing faces in cases where a substantial portion is hidden encounters challenges, significantly impacting the system's accuracy.

The experimental results we obtained in the study were also compared with the results of similar studies in the literature (**Table 3**).

Table 3 Evaluation of performed experiments

State	Accuracy (%)
With One Person's Face Masked (This Study)	85
One Person Face Masked (Karadağ, 2020)	84.5
While One Person's Face Is Masked (Özdemir, 2019)	80
While One Person's Face Is Masked (Atasoy, 2018)	80

As shown in Table 3, the success rate of 85% obtained as a result of this study is slightly lower than the success rate of 84.5% in the study of Karadağ (2020). While Karadağ's study showed that he was able to achieve slightly higher accuracy in recognizing masked faces, this study still achieved an acceptable level of recognition success of 85%. On the other hand, Özdemir's (2019) study also achieved 80% success in recognizing masked faces. This is a result quite similar to the success rate in this study. Similarly, 80% success was achieved in Atasoy's (2018) study. While these two studies had similar success levels in recognizing masked faces, this study focused on more complex situations and achieved an 85% success rate. In conclusion, this study performs competitively in the recognition of masked faces when compared to other similar studies. While the work of Karadağ has slightly higher success, a similar level of success is shown with the work of Özdemir and Atasoy. This comparison helps us to evaluate the masked face recognition capabilities of the study from a broader perspective.

This series of experiments was carried out to evaluate the performance of different methods in face recognition and mask classification problems. The experimental results reveal the advantages and difficulties of each method. The effectiveness of pre-trained models in face recognition and the success of the Haar Cascade method in mask detection are interesting. This study provides important information that can be used in a wide area from security applications to biometric recognition systems.

7. CONCLUSION

The results of this study reveal the limitations of traditional face recognition systems when individuals wearing masks have their faces covered. In particular, the fact that the faces of people wearing masks are largely covered significantly reduces the accuracy of existing systems. The research presents an approach that enables real-time recognition of masked faces using the Haar Cascade method and combines this with MongoDB database integration. Experiments show that the proposed method is effective even under different lighting conditions and is suitable for real-world applications. It is considered that this study will contribute to the literature within the scope of its potential to offer a more effective solution, especially by addressing the difficulties in mask detection.

KAYNAKÇA

Araujo, J. M. A., de Moura, A. C. E., da Silva, S. L. B., Holanda, M., de Oliveira Ribeiro, E., da Silva, G. L. (2021, June). Comparative performance analysis of NoSQL Cassandra and MongoDB databases. In 2021 16th Iberian Conference on Information Systems and Technologies (CISTI) (pp. 1-6). IEEE.

Atasoy, N. A., Tabak, D. (2018). Destek vektör makineleri kullanarak yüz tanıma uygulaması geliştirilmesi. Engineering Sciences, 13(2), 119-127.

Banker, K., Garrett, D., Bakkum, P., Verch, S. (2016). MongoDB in action: covers MongoDB version 3.0. Simon and Schuster.

Boicea, A., Radulescu, F., Agapin, L. I. (2012, September). MongoDB vs Oracle database comparison. In 2012 third international conference on emerging intelligent data and web technologies (pp. 330-335). IEEE.

Ciotti, M., Ciccozzi, M., Terrinoni, A., Jiang, W. C., Wang, C. B., Bernardini, S. (2020). The COVID-19 pandemic. Critical reviews in clinical laboratory sciences, 57(6), 365-388.

Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM) (pp. 210-214). IEEE.

da Rocha França, W. (2015). MongoDB data modeling. Packt Publishing Limited.

Daşdemir, Y., Kara, B. C. (2019). Farklı iş yükleri altında NoSQL sistemlerinin performans analizi. Bitlis Eren Üniversitesi Fen Bilimleri Dergisi, 8(4), 1466-1477.

Mongo, D. B. (2015). Mongodb.

Grudin, M. A. (2000). On internal representations in face recognition systems. Pattern recognition, 33(7), 1161-1177.

Karadağ, N., Çetinkaya, A., & Aydın, H. (2020). Yerel İkili Desenler Histogramları ile Covid-19 Tanılı Kişiler Üzerinde Kimlik Analizi ve Bildiri Sistemi. Gazi Mühendislik Bilimleri Dergisi, 6(3), 172-183.

Kekül, H., Bircan, H., Arslan, H. (2018). Yüz tanıma uygulamalarında özyüzler ve yapay sinir ağlarının karşılaştırılması. Uluslararası Yönetim Bilişim Sistemleri ve Bilgisayar Bilimleri Dergisi, 2(1), 51-59.

Kirby, M., Sirovich, L. (1990). Application of the Karhunen-Loeve procedure for the human faces. IEEE Trans. Pattern Analysis and Machine Intelligence, characterization, 12(1), 103-108.

Kolakowski, (2020, December, 3), 10 Most Popular Programming Languages on GitHub, GitHub Language Rankings, 2018-2020 (Accessed date:

October 19, 2021) https://insights.dice.com/2020/12/03/10-most-popular-programming-languages-on-github/

Kortli, Y., Jridi, M., Al Falou, A., Atri, M. (2020). Face recognition systems: A survey. Sensors, 20(2), 342.

Lemenkova, P. (2020). Python libraries matplotlib, seaborn and pandas for visualization geospatial datasets generated by QGIS. Analele stiintifice ale Universitatii" Alexandru Ioan Cuza" din Iasi-seria Geografie, 64(1), 13-32.

Lutz, M. (2010). Programming Python: powerful object-oriented programming. " O'Reilly Media, Inc.".

Mamak, U., Konyar, M. Z., Solak, S., Uçar, M. H. (2020). Gerçek zamanlı yüz tanıma tabanlı personel kontrol ve takip sistemi tasarımı. Avrupa Bilim ve Teknoloji Dergisi, (19), 497-504.

Mundial, I. Q., Hassan, M. S. U., Tiwana, M. I., Qureshi, W. S., Alanazi, E. (2020, September). Towards facial recognition problem in COVID-19 pandemic. In 2020 4th International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM) (pp. 210-214). IEEE.

Ngo, H. T., Rakvic, R. N., Broussard, R. P., Ives, R. W. (2009, May). An FPGA-based design of a modular approach for integral images in a real-time face detection system. In Mobile Multimedia/Image Processing, Security, and Applications 2009 (Vol. 7351, pp. 83-92). SPIE.

Özdemir, R., & Mehmet, K. O. Ç. (2019). Yeni bir veri kümesi (RidNet) kullanarak kontrolsüz ortamda yüz ifadesi tanımanın derin öğrenme yöntemleri ile iyileştirilmesi. Bilecik Şeyh Edebali Üniversitesi Fen Bilimleri Dergisi, 6(2), 384-396.

Parmar, D. N., Mehta, B. B. (2014). Face recognition methods & applications. arXiv preprint arXiv:1403.0485.

Saran, M., Saran, N. (2019). Cassandra ve MongoDB NoSQL Veri Tabanlarının Karşılaştırmalı Güvenlik Analizi. Uluslararası Bilgi Güvenliği Mühendisliği Dergisi, 5(2), 1-11.

Terzopoulos, D., Waters, K. (1990, January). Analysis of facial images using physical and anatomical models. In Proceedings Third International Conference on Computer Vision (pp. 727-728). IEEE Computer Society.

Rodriguez, Y., Marcel, S. (2006). Face authentication using adapted local binary pattern histograms. In Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part IV (pp. 321-332). Springer Berlin Heidelberg.

Taylan, E. (2014), ABD'deki üniversitelerde en çok öğretilen programlama dili: Python Erişim Tarihi:19-10-2021.

Van Natta, M., Chen, P., Herbek, S., Jain, R., Kastelic, N., Katz, E., Vattikonda, N. (2020). The rise and regulation of thermal facial recognition technology during the COVID-19 pandemic. Journal of Law and the Biosciences, 7(1), lsaa038.

Vansh, V., Chandrasekhar, K., Anil, C. R., Sahu, S. S. (2020). Improved face detection using YCbCr and Adaboost. In Computational Intelligence in Data Mining: Proceedings of the International Conference on ICCIDM 2018 (pp. 689-699). Springer Singapore.

BİLGİSAYAR BİLİMLERİ VE TEKNOLOJİLERİ DERGİSİ
JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGIES
http://dergipark.org.tr/tr/pub/bibted

Araştırma Makalesi

HİBRİT BULUT: AWS NEDİR NASIL KULLANILIR

Büşra BAŞ[1]

[1]*Mehmet Akif Ersoy Üniversitesi, Sosyal Bilimler Enstitüsü, Yönetim Bilişim Sistemleri, Burdur, Türkiye*

Anahtar Kelimeler:
Bulut bilişim
Sanal tabanlı sunucu
Amazon web servisi
S3
EC2

ÖZ

Dijital dünyada depolama, verilerin saklanması işlemidir. Önceleri yerel belleklerde tutulan veriler, son zamanlarda sunuculardа tutulmaktadır. Verilerin tutulduğu sunucuya bulut adı verilmektedir. Dünyada birçok bulut depolama sunucusu bulunmaktadır. Bu sunuculardan biri olan Amazon Web Services(AWS) dünyada büyük bir pazara sahiptir. Ancak ülkemizde kullanım oranı düşüktür. Bunun nedeni kullanıcıların sistemi tanımaması ve güvenlik tereddütleri olabilir. Çalışma kapsamında; "bulut bilişim nedir", "dalları nelerdir", "hibrit bulut neden kullanılır", "Amazon Web Services (AWS) nedir", "Amazon Web Services (AWS)' de depolama nasıl oluşturur", "hangi sunucu hangi görevi üstlenmektedir", "işletmelere olan katkısı nelerdir" gibi sorular cevaplanmaya çalışılmıştır. Araştırma sonucunda Amazon Web Services (AWS)' de bulunan depolama sunucularından en çok Elastik Bilişim Bulutu (EC2) ve Basit Depolama Hizmeti (S3)'nin tercih edildiği saptanmıştır. Elastic Bilişim Bulutu (EC2) daha büyük ve daha işlevsel veri tutarken, Basit Depolama Hizmeti(S3) daha basit yapılı verileri tutmaktadır. Amazon Web Services (AWS) veri tabanı kullanımı yaparken veriler yüklenirken küçük boyutlu ve küçük boyutlu kalacak ise İlişkisel Veri Tabanı Hizmeti (RDS) kullanılır. Yükleme yapılırken büyük boyutlu veri girişi sağlanıyorsa veya veri gittikçe büyüyecek ise Dinamo Veri Bulutu (Dynamo DB) modeli kullanılması önerilmektedir.

HYBRİD CLOUD: WHAT İS THE AWS HOW TO USE

Keywords:
Cloud computing
Virtual based server
Amazon web services
S3
EC2

ABSTRACT

Digital storage is the process of storing data. The data, which was previously kept in local memories, has recently been kept on servers. The virtual base where the data is kept is called the cloud. There are many cloud-storage servers in the world. Amazon Web Services (AWS), has a large market in the cloud-storage world. However, its usage rate is low in our country. Scope of work; "what is cloud computing", "what are its branches", "why use hybrid cloud", "what is AWS", "how to create storage in AWS", "which server undertakes which task", Questions such as "what is its contribution to businesses" were tried to be answered. As a result of the research, it has been determined that Elastic Computing Cloud (EC2) and Simple Storage Service (S3) are mostly preferred among the storage servers in AWS. EC2 holds larger and more functional data, while S3 holds data with a simpler structure. When using AWS database, Relational Database Service (RDS) is used if the data will remain small in size while being loaded. It is recommended to use the Dynamo Data Cloud (Dynamo DB) model if large-scale data entry is provided while uploading, or if the data will grow gradually.

*Sorumlu Yazar

*(busrabas1998@icloud.com) ORCID 0009-0000-9378-0971

e-ISSN: 2717-8579

Geliş Tarihi: 06/03/2023; Kabul Tarihi: 09/08/2023

1. GİRİŞ

Dünya genelinde insanlar, geçmişten bugüne kadar hep depolamaya ve veri tutmaya ihtiyaç duymuştur. Veri tutmanın ilk örneklerinden olan veresiye defteri buna en güzel örnektir. Teknoloji gelişmeden önce veri depolamak için kullanılan defter, teknoloji geliştikçe yerini gerek cihazlara gerek ise dijital aletlere devretmiştir. Geçmişte kullanılan CD yerini flaş diske, flaş disk yerini hard diske, hard disk ise yerini bulut bilişime bırakmıştır. Bahsi ilk 1950 yılında geçmiş olsa da özel depolama için ilk kullanılan bulut servisi amazon S3, 2006 yılında hizmete girmiştir(Mathew, 2014). Bulut sistemi insanların işlerini kolaylaştırmak ve maliyeti düşürmek için tasarlanmış veri ambarı bütünüdür. Özellikle işletmeler için büyük nimet sayılan bulut, işletmeleri büyük donanım maliyetlerinden kurtarmıştır. Bulut adı verilmesinin temelinde gözle görülmeyen bir ağ olması yatmaktadır. Bulut denilince akla yeni bir kavram olarak gelse de aslında internet ilk kullanılmaya başlandığında birçok servis bulut üzerinden sağlanmaktaydı fakat kullanıcılar tarafından bilinmiyordu. Bulut bilişim, katmanlarında birçok servis, altyapı ve platform barındırmaktadır(Okutucu, 2012). Bulut bilişimin hizmet alanında dört faklı tür kullanılmaktadır(Şanlı, 2011). Bu hizmet modellerine aynı zamanda farklı dağıtım modelleri de denilmektedir. Dört farklı dağıtım modelleri kısaca tanıtılmak istenirse;

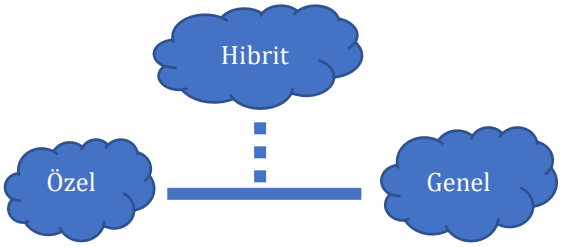
1. Genel Bulut (Public Cloud): İngilizce karşılığı halka açık bulut demek olsa da genel bulut, bir bulutun parçalara ayrılarak diğer işletmeler ile ortak kullanım sağlayan bulut türüdür. Fakat bu bir işletmenin bilgilerini diğer işletmeler tarafından görülebilmesi demek değildir. Ortak kullanılmakta fakat kullanıcılar, diğer kullanıcıların verilerini görememektedir.

2. Özel Bulut (Private Cloud): Adından da anlaşılacağı üzere, özel olarak veri merkezlerinin içinde çalışır ve yönetir.

3. Topluluk Bulutu (Community Cloud): Genel Bulut'un güvenliğini ve Özel Bulut'un maliyetini azaltmakta olan bulut biçimidir. Birden fazla şirketle ortak kullanılan hizmetleri kapsayan sunucudur.

4. Hibrit/Karma Bulut (Hybrid Cloud): İki ve daha fazla bulut sistemini içinde bulunduran bulut sağlayıcısıdır. Topluluk bulutunda şirketlerin kullandığı hizmetleri kapsayan sunucu iken hibrit bulut, bir şirketin isteklerine göre kullandıkları bulut türlerine denir.

Bunlar içinde bulunan hibrit (karma) bulut bilişim, birden fazla bulut modelini bir arada tutan programdır. Açık olarak ifade etmek istenirse hibrit bulut; özel bulut ve genel bulut hizmetlerinin birleştiği bir bulut yapısıdır. Hibrit bulutu aşağıdaki şekil ile anlamak daha mümkün olacaktır.



Şekil 1. Hibrit Bulut Bilişim

Amazon Web Servisi, insanların hayatlarını kolaylaştırmak ve maliyetleri düşürmek için yapılmış sunuculardır. Bu çalışmanın amacı Amazon web servislerini tanıtmak ve kapsamlı bir araştırma yapmaktır.

Literatürdeki çalışmalar genel olarak bulut bilişim, hibrit bulut bilişim ve Amazon'un kullanıldığı projeler ayrı olarak araştırılmış, amazon için karşılaştırmalara yer verilmiş fakat kullanım bilgisine yer verilmediği gözlemlenmiştir. Bu çalışmaya ilişkin araştırmalarda bazı benzerlikler vardır fakat kullanımı konusunda benzerliğe rastlanılmamıştır. Çalışmanın bir diğer benzeri Microsoft Azure için yapılmış olup Amazon için kapsamlı bir çalışma görülmemiştir. Padhy vd. tarafından yazılan "Windows Azure Paas Cloud: An Owerview" adlı çalışmasından örnek alınarak ve üstüne katılarak bir çalışma yapılmıştır. Amazon web servisleri konusunda bu formatta bir çalışmaya rastlanılmamış ve kapsamlı olarak Amazon çalışması yapılmıştır. Bu çalışmanın amacı, hibrit bulut bilişim nedir? Hibrit bulut bilişim ile oluşturulan Amazon web servisleri nedir, bu servislerde depolama nasıl yapılır, özellikleri nelerdir sorusunu yanıtlayabilmek için yapılmış tarama çalışmasıdır.

2. KAVRAMSAL ÇERÇEVE

2.1. Terim Listesi:

Cloud : Bulut
DNS : Etki alanı isimlendirme hizmeti
Host-based : Ev sahibi tabanlı
SOA : Servis tabanlı mimari
SaaS : Bulut yazılım hizmeti
VPN : Sanal özel ağ
Auto Scalin : Ram arttıkça, kapasiteyi arttıran sistemdir
AWS : Amazon Web Servisi
EC2 : Elastik Bilişim Bulutu
S3 : Basit Depolama Hizmeti
SES : Basit E- posta Hizmeti
EBS : Elastik Blok Depolama
CND : İçerik Dağıtım Ağı
Private : Özel
Public : Genel
EFL : Elastik Dosya Sistemi

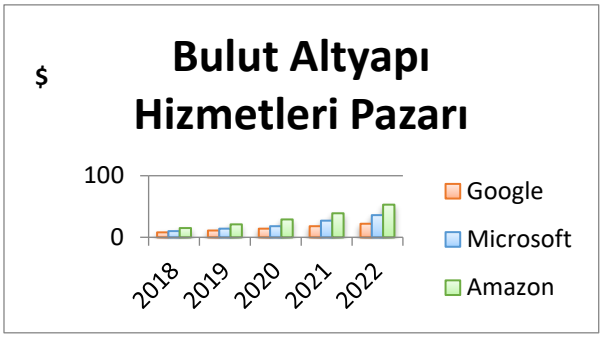
3. AMAZON WEB HİZMETİ

Amazon web servisleri (AWS) 2006 yılında bilgi teknolojileri altyapısında bulut hizmeti vermeye başlamıştır. Bulut, bilgi işlemenin en önemli faydalarından biri olan ön sermayeyi değiştirme fırsatıdır. İşletmelerinize göre değişen altyapı ve donanım maliyetlerini düşürmektedir(Mathew, 2014). Servisi kullanan işletmelerin, maliyet lideri konumuna gelerek sektörde olduğundan daha yüksek paya erişebilmektedir. AWS hayatımıza girmeden önce işletmeler donanım için çok fazla sermaye harcamaktaydı. Depolama için ayrılan sermaye ve yer oldukça fazla düşünülmekteydi. AWS hayatımıza girerek, düşük maliyet ile çok fazla veriyi kolaylıkla depolamayı insanoğluna öğretmiştir.

Bugün AWS son derece güvenilirdir. Düşük maliyetli altyapıyı 190 ülkede, yüzbinlerce işletmeye güç sağlamaktadır. Amazon Basit E- posta Hizmeti (Amazon SES), kuruluşlar ve geliştiriciler için son derece ölçeklenebilirdir. Uygun maliyetli toplu ve işlem e- posta gönderme hizmetidir. Amazon SES, şirket içi bir e- posta çözümü oluşturmanın veya bir üçüncü taraf e- posta hizmetini lisans almanın, kurmanın ve çalıştırmanın karmaşıklığını, masrafını ortadan kaldırmaktadır. SES, diğer AWS hizmetleriyle uyum sağlayarak Amazon EC2 gibi hizmetlerde barındırılan uygulamalardan e- posta göndermeyi kolaylaştırmaktadır. Amazon SES, uzun vadeli taahhüt, minimum harcama veya müzakere gerektirmemektedir. Kuruluşlar ücretsiz bir kullanım katmanı kullanabilmektedir. Ardından gönderilen e- posta sayısı için düşük ücretler ve veri aktarım için düşük ücretlerinden yararlanılmaktadır. Pazarlama ve işlem mesajları gönderebilmek için büyük ölçekli e-posta çözümleri oluşturmak genellikle karmaşık ve maliyetli olmaktadır. Amazon bu yönü ile diğer kuruluşlara meydan okumaktadır. Başarılı bir şekilde teslim edilen e- postaların yüzdesini optimize etmek için kuruluşların e-posta sunucusu yönetimi ve ağ yapılandırması ile ilgilenmektedirler. Ayrıca titiz internet servis sağlayıcısını karşılamaları gerekmektedir. E- posta içeriği Standartları gereği uyulması gereken kurallar barındırmaktadır. Önemli ön maliyetlerin yanı sıra birçok zorluk ile karşılaşılmaktadır. Amazon SES bu zorlukları ortadan kaldırmaktadır. Kuruluşların Amazon.com'un kendi geniş ölçekli müşteri tabanına hizmet vermek için oluşturduğu uzun yıllara dayanan deneyimden ve gelişmiş e- posta altyapısından yararlanmasını sağlamaktadır(Mathew, 2014). Amazon, müşterileri ile iletişime geçebilmek için bir e-posta sağlayıcı oluşturmuştur. Szul'a(2022) göre Amazon Basit E-posta Hizmeti (SES), Amazon.com'un kendi müşteri tabanına hizmet vermek için geliştirdiği güvenilir ve ölçeklenebilir altyapı üzerine kurulmuş uygun maliyetli bir e- posta hizmetidir. Amazon SES ile işleme dayalı bir e posta gönderilebilmektedir. AWS, bu kapasiteyi minimum maliyet sağlamada yardımcı olmaktadır. Bu fayda sayesinde iş yükleri azalmakta ve odaklanıp farklı fikirler inşa etmektedir.

Wall vd.(2011)'e göre Amazon web hizmetleri, CPU'lar RAM, depolama, işletim sistemi ve ağ iletişimi dâhil olmak üzere gerekli bilgi işlem ortamını sağlamaktadır. Amazon, hizmet altyapı(IaaS) örneği olarak gösterilmektedir. IaaS, projeleri kalıcı olmadan tasarlamak için daha fazla esneklik sunması tarafından çok popülerdir. Hesaplamalı projelerde AWS'in üç ürünü kullanılmaktadır. Bunlar; Elastik Blok Depolama (EBS), Basit Depolama Hizmeti (S3) ve EC2'dur.

Bermudez vd,(2013)'ne göre Amazon Web Servisleri, son yıllarda, uzak veri merkezlerinde bilgi işlem, depolama ve boşaltma yetenekleri sağlamaktadır. Müşterilere donanım yönetimini sanallaştırılarak maliyetleri düşürme fırsatı sunan bulut tabanlı hizmetlerin büyümesinde tanık olunmuştur. Bu panoramada lider konum, Amazon Web Servis (AWS) adlı geniş bir bulut tabanlı hizmetler yelpazesi sunmaktadır. En iyi bilinen Amazon bulut hizmetleri; içerik Dağıtım Ağı(CDN) olan CloudFront, Elastik Bilişim Bulutu (EC2) ve Basit Depolama Hizmeti (S3)'dir. AWS bir altyapı sağlayıcısını temsil etmektedir. EC2 ve S3, hizmet olarak altyapı ürünlerine karşılık gelmektedir. Başka bir değişle sanallaştırma aracılığı ile depolama ve işleme kapasiteleri gibi büyük bir bilgi işlem kaynakları seti, müşterilerin talebini karşılamak için bölünebilir, atanabilir ve dinamik olarak boyutlandırılabilmektedir. Müşteriler, hizmetlerini kendi donanımlarını, altyapılarını kurma ve yönetme, maliyet ve risklerini taşımadan sunmayı amaçlayan şirketler tarafından temsil edilmektedir. Dropbox ve Netflix gibi birçok başarılı şirket, AWS'e çok güvenmekte ve kullanmaktadır.



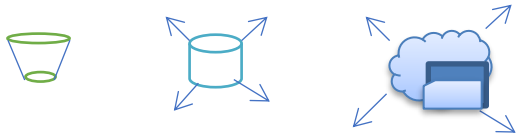
Şekil 2. Amazon'un Dünyadaki Pazar Payı(Synergy Research Group)

3.1. AWS Hizmet Alanları

İşlem Depolama Veri Tabanı Ağ İletişimi ve İçerik Teslimi

Şekil 3. AWS Depolama Hizmetleri

3.2. AWS Depolama Hizmetleri



Amazon Simple Storage Service System (Amazon S3) | Amazon Elastic BlockStore (Amazon EBS) | Amazon Elastic File (Amazon EFS)

Şekil 4. AWS Hizmet Alanları

4. AMAZON SİMPLE STORAGE SERVİCE (Amazon S3)

• İnternet üzerinden istenilen türde veriyi depolayabilir ve erişim sağlayabilmektedir.
• %99,999999999 dayanıklı olduğu test edilerek kanıtlanmıştır.
• Kullanıcılara uygun maliyetli, geniş depolama sınıfları sunmaktadır.
• Kullanım yeri olarak, genellikle telefondaki fotoğrafların depolandığı servis birimidir.
• Her obje, aynı bölge içinde 3 ayrı veri merkezinde yedeklenebilmektedir. Bu sebepten dolayı, 11,9'luk yüksek erişim sunulmaktadır.
• Dosyaları düzenli klasörler halinde depolamaktadır.
• Birçok farklı kullanım örneğini desteklemektedir.
• Farklı depolama ve klasörleme biçimi kullanıldığı için silinmeye karşı üstün koruma sağlamaktadır(Vural, 2022).

Suzl'a(2022) göre S3, Amazon basit depolama hizmeti veya kısaca S3, geliştiricilere ve bilgi teknolojileri ekiplerine güvenli nesne veya veri depolama alanından tasarruf etme olanağı sağlamaktadır. Amazon EC2'den veya web üzerindeki herhangi bir yerden herhangi bir zamanda herhangi bir miktarda alınan veriyi depolamak ve almak için kullanılabilen basit web hizmeti ara yüzü ile kullanımı kolaydır. Amazon S3 ile yalnızca kullanılan depolama alanı için ödeme yapılmaktadır. Minimum ücret ile çalışmaktadır ve kurulum maliyeti yoktur. Amazon S3 Glacier ise veri arşivleme ve uzun vadeli yedekleme için güvenli, dayanıklı ve son derece düşük maliyetli bir bulut depolama hizmetidir. Müşteriler, büyük veya küçük miktarlardaki verileri, şirket içi çözümlere kıyasla önemli ölçüde tasarruf sağlayan aylık gigabayt başına 0,004 ABD Doları gibi düşük bir fiyata güvenilir bir şekilde depolama sağlamaktadır.
Wall vd.(2011) göre S3, yüklenen verileri internet üzerinden kolayca kullanabilir hale getiren kalıcı ve son derece güvenilir bir depolama sistemidir. S3 güvenirliği sağlamak için Amazon'un birden çok, farklı coğrafi altyapısında bulunan hizmetlerde saklamaktadır. Yani yedek verilerin yedeğini alarak kaybolmaları en aza indirmektedir. Böylece AWS nesneleri veya verileri için %99,999999999 dayanıklılık ve %99,99 kullanılabilirlik sunmaktadır. Amazon büyük veri kümelerini S3 üzerinden genel klasörler ile kullanılabilir hale getirmektedir.
Mathew(2014)'e göre S3, internet için depolamadır. Geliştiriciler, web ölçeğinde bilgi işlemi kolaylaştırmak için tasarlanmıştır. Amazon S3, herhangi bir zamanda, herhangi bir miktarda veriyi web üzerinde herhangi bir yerden depolamak ve almak için kullanılabilen basit bir web hizmetleri ara birimi sağlamaktadır. Amazon S3'te depolanan nesnelerin kapsayıcısına Amazon S3 Bucket adı verilmektedir. S3, herhangi bir geliştiricinin Amazon'un sunduğu yüksek düzeyde ölçeklenebilir, güvenilir, hızlı ve ucuz altyapıya erişmesini sağlamaktadır. S3'te sağlanan hizmet, ölçeğin faydalarını en üst düzeye çıkarmayı ve bu faydaları geliştiricilere aktarmayı amaçlamaktadır. Amazon S3 Glacier ise, veri arşivleme ve yedekleme için güvenli, dayanıklı depolama sağlayan son derece düşük maliyetli bulut tabanlı bir depolama hizmetidir. Kullanımı daha rahattır. Müşterilerin aylık gigabayt başına 0,01 ABD Doları gibi bir maliyetle büyük veya küçük miktarda verileri güvenilir bir şekilde depolayabilmektedir.

4.1. Amazon Elastic Block Store (Amazon EBS)

Amazon EC2 makinalarına takılan diskler için kullanılan ifadedir. Application server kurmak için altyapı sağlamaktadır. Özellikleri aşağıda yer verilmiştir.

Özellikler:

• Yüksek performanslı blok depolama sağlamaktadır.
• %99,999 erişebilirlik sağlamaktadır ve erişebilirlik alanı içinde çoğaltılmaktadır.
• Ölçek dakikalar içinde arttırılıp, azaltılabilmektedir.
• Fiyat ve performans optimizasyonu için dört birim türü seçeneği sunmaktadır. Bunlar:

Genel amaçlı SSD, tedarik edilmiş IOPS SSD, aktarım hızı için optimize edilmiş HDD, seyrek erişimli HDD.

Aynı anda 2 EC2 makineyi tek bir diski okuyup yazması isteniyor ise bu EBS ile mümkün

olmayacaktır. Eğer böyle bir şey isteniyor ise Elastic File System (EFS) ile mümkün olabilecektir(Vural, 2022). EBS birimi, çalışan örneklere eklenebilen, USB sürücüye benzeyen bir depolama aygıtıdır. Yaşanılan bu dönemde boyutları 1GB ile 1TB arasında değişebilmektedir. EBS birimleri yedek verilerin de yedeğini almaktadır. Bu sayede veri kaybolmalarını en aza düşürmektedir. Veri dayanıklılığı yaklaşık %99,7'dir. Son yedeklemeleri Amazon S3'e yüklemektedir(Wall vd,2011). Amazon Elastik Block Store (EBS), Amazon EC2 bulut sunucularıyla kullanım için blok düzeyinde depolama birimleri sağlamaktadır. Amazon EBS'si ile birimler ağa bağlıdır. Bir verinin ömründen bağımsız olarak devam etmektedir. Amazon EBS, çalışan bir Amazon EC2 bulut sunucusuna eklenebilen ve bulut sunucusu içinde bir cihaz olarak gösterilebilen yüksek düzeyde kullanılabilmektedir. Yüksek düzeyde güvenilir, öngörülebilir depolama birimleri sağlamaktadır. Amazon EBS, bazı özellikler için uygundur. Bu özellikler şunlardır: Bir veri tabanı, dosya sistemi veya ham blok düzeyinde depolamaya erişim gerektiren uygulamalar için uygundur(Mathew, 2014).

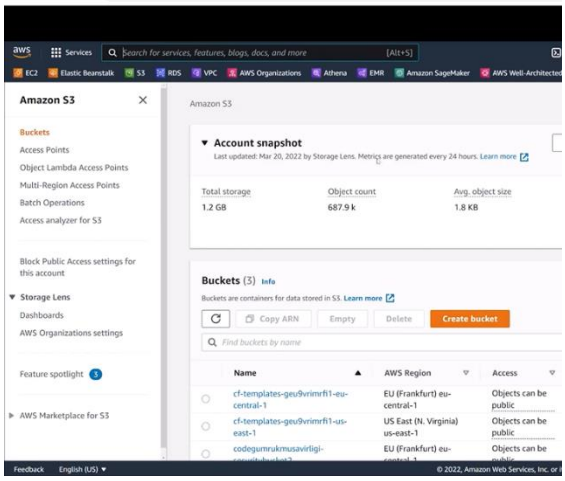
4.2. Amazon Elastic File System (EFS)

- Basit, ölçeklenebilir ve tam olarak yönetilebilen dosya sistemidir.
- İstek üzerine ve uygulamaların çalışmasını etkilemeden petabayt ölçeğinde genişletilebilme özelliğine sahiptir.
- Verilerin birden çok erişebilir alanda depolama sağlamaktadır.
- Çoklu okuma-yazma ortamı sağlamaktadır.
- Dinamik esneklik sağlamaktadır.
- Uygun maliyetli kullanım hizmeti sağlamaktadır.
- Paylaşım yapılan dosya için depolama alanı sağlamaktadır.

5. AMAZON'DA DEPOLAMA OLUŞTURMA

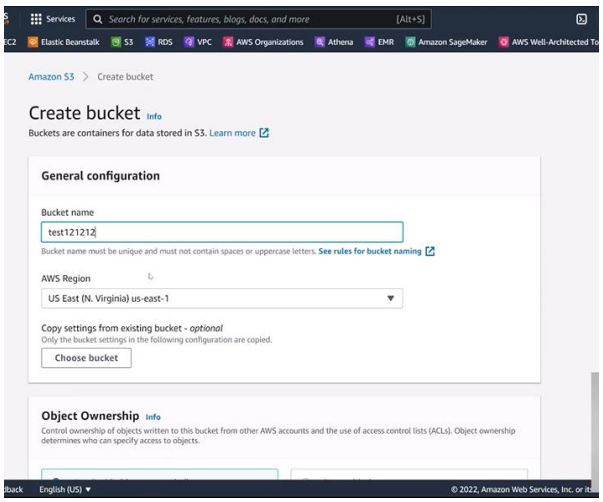
Amazon'da bir depolama oluşturabilmek için ilk önce Amazon'un ana sayfası olan AWS'e giriş yapılması gerekmektedir. AWS' e giriş yapıldıktan sonra arama kısmına "S3" yazılarak arama başlatılır. Açılan sayfada yer alan "create bucket" butonuna tıklanarak bir kova yaratılması gerekmektedir. Butona tıklama yapıldıktan sonra açılan sayfada öncelikli olarak bucketımıza/kovamıza bir isim vermemiz beklenmektedir. İsim verirken, ismi eşsiz olarak seçildiğinden emin olunması gerekmektedir. Bunun sebebi ise olası veri kayıplarına sebep olmamak içindir. Açılan sayfada ismi verdikten sonra aynı sayfada yer alan "AWS Region" kısmında hangi bölgede kurulum yapılacağı seçilmelidir. Türkiye'de daha iyi hizmet alabilmek için "EU (Frankfurt) eu-centrol-1" seçeneği işaretlenmelidir. Object Ownership kısmından sadece kovayı oluşturan kişinin yönetebilmesi için "ACLS Enabled" seçeneğini ardından "Bucked Owner Preferred" seçeneği işaretlenmelidir. Eğer kovaya hiçbir yerden ulaşılması istenmiyor ise "Block All Public Access" seçeneği işaretlenmelidir. Dışarıdan erişilmesi isteniyor ise bu seçenek işaretlenmemelidir. İşaretli ise de kaldırılmalıdır. Aynı sayfanın altında yer alan bilgilendirme kabul edilmelidir. Bucket Versioning kısmında "enable" sekmesi seçilmelidir. Bunun sebebi ise bir objenin, birden çok versiyonunu depolaması ve bu objenin birden çok versiyonunun aynı isimle farklı sistemlerde kayıt olması isteniyor ise "enable" seçeneği işaretlenmelidir. Tags-Optional kısmında yer alan bölüme klasör halinde açılım yapabilmek için key kısmına bir sözcük, value kısmına bir isim girilmesi istenmektedir. Örnek vermek gerekir ise, Key kısmına "amac" value kısmına "fotograf yedegi" yazarak add tag butonuna tıklanmaktadır. Türkçe karakter kullanılmamasına özen gösterilmesi gerekmektedir. Default encryption kısmını "enable" seçildiği takdirde iki kısım karşımıza çıkmaktadır. 1. kısım "Amazon S3-Managed Keys (SSE-S3)" bu kısımda yöneticinin herhangi bir anahtar atamasını sağlamaktadır. 2. kısımda ise "AWS Management Service Key (SSE-KMS)" Amazon servis hizmetlerinin bir anahtarını kullanabilmek için seçilmesi gerekmektedir. Bu şık seçildiği anda 3 ayrı sekme ortaya çıkmaktadır. İlk sekmede "aws managed key (AWS S3)" yani key SSE'nin ürettiği bir anahtar kullanmak için seçilebilecek bir seçenektir. Açılan 2. sekme ise "Choosse From Your AWS KMS Keys" yani daha önce seçilmiş anahtarı getirerek onu seçmektedir. Açılan 3. sekmede ise kullanıcının özgün bir anahtar oluşturması beklenmektedir. "Enter KMS Key ARN" butonuna tıklanarak bucket tamamlanabilmektedir. Oluşturulan bucket'a tıklanıldığında karşımıza objects, properties, permissons, metrics, management, acces points sekmeleri çıkmaktadır. Objects sekmesine tıklanıldığında bu objenin ne olduğunu gösteren sekme türüdür. Proporties kısmında bucket'ın özellikleri görülebilmektedir. Diğer bir sekme olan Permissons sekmesi ise bu bucket'a kimlerin erişebileceğini ayarlayabilen kısımdır. Metrics sekmesinden kaç obje bulunmaktadır, boyutları ne kadardır gibi soruları cevaplayan sekmedir. Management sekmesi ise kuralları belirlemektedir. Ürünleri AWS'e göre yüklemek ise şu şekilde gerçekleştirmektedir. Oluşturulan bucket'a giriş sağlanır, object sekmesinden update butonuna tıklayarak dosya, fotoğraf vb. gibi veriler yüklenebilmektedir. Yüklenilen veriyi silmek için ise, silinmesi istenen dosyanın üzerine tıklanır ve delete komutuna tıklanır. Ekrana gelen açıklama satırına "delete" yazılarak onay tuşuna tıklanır. Ayrı klasörde

veri tutabilmek için ise, object sekmesine girilmeli daha sonra alt kısımda yer alan folder butonuna tıklayarak, folder name kısmına klasöre vermek istenilen isim yazılmalıdır. Daha sonra server-side encryption kısmını "disable" seçerek oluşturulmaktadır(Vural, 2022).



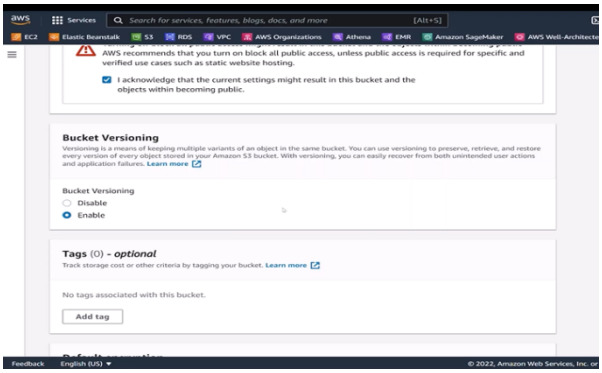
Şekil 5. Amazon S3 Ana Sayfa

Yukarıda yer verilen ekran görüntüsünde Amazon AWS sayfasında S3 hizmetinin açılmış olduğu sayfaya yer verilmiştir. Sayfada bulunan create bucket butonu, bizim yeni bir bucket yani kova oluşturmamızı sağlayan butondur. Alt kısımda bulunan name tablosunda oluşturulan kovalar yer almaktadır.



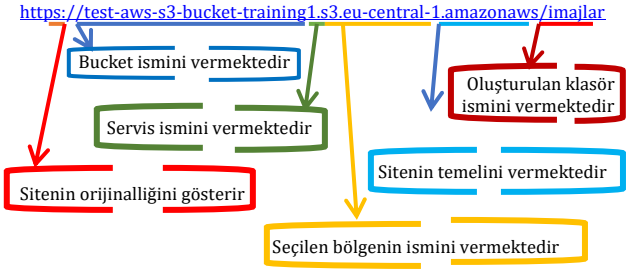
Şekil 6. Amazon S3 Create Bucket

Yukarıda görülen ekranda create bucket butonuna tıklandığı zaman açılan sayfa görülmektedir. Bucket name adlı satırına eşsiz isim atadığı yerdir. AWS Region kısmı ise sanal makineyi nerede kuracağımızı seçeceğimiz alandır.



Şekil 7. Amazon S3 Create Bucket Version

6. AMAZON LİNK GÜVENİRLİĞİ



Şekil 8. Amazon Güvenli Link

Yukarıda yer alan linkte, güvenilir bir Amazon linki nasıl olur sorusunun cevabı yer almaktadır. Amazon bizlere bir milyon dosyayı depoladığımızda, bir yıl içinde maksimum bir dosyanın kaybolacağının teminatını vermektedir.

6.1. Amazon Web Hizmeti'nde Güvenlik

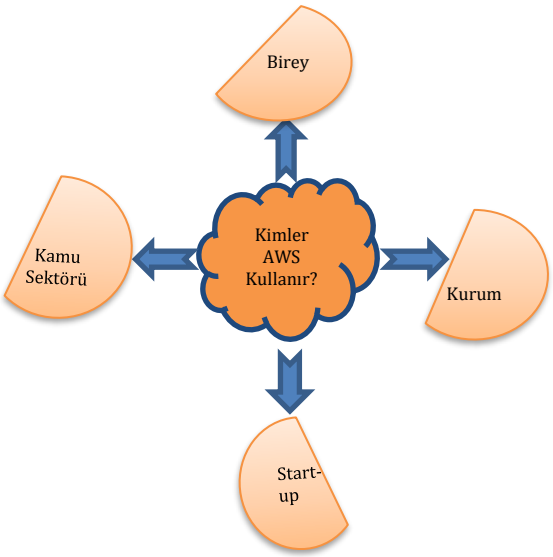
AWS bulut altyapısı, en esnek, güvenli bulut bilgi işlem ortamlarıdır. Günümüzde en iyi şekilde servis hizmetleri olarak mevcuttur. Son derece ölçeklenebilir olan AWS, müşterilerin uygulamalarını ve verilerini hızlı bir şekilde devreye almayı sağlayan son derece güvenilir bir platformdur. İnsanlar bulut kullanmak istediklerinde, en çok başlarını ağrıtan konu güvenlik olarak ortaya çıkmaktadır. AWS güvenlik açığı oluşturmaması adına birinci sınıf, son derece güvenli veri merkezlerini barındırmaktadır. Son teknoloji ürünü olan elektronik gözetim ve çok faktörlü erişim kontrol sistemlerini kullanmaktadır. Veri merkezleri, eğitimli güvenlik görevlileri tarafından yani yeni adıyla siber güvenlik uzmanları tarafından 7/24 çalıştırılır ve erişime yetkilendirilir. Çevre sistemlerini en aza indirecek şekilde tasarlanmıştır. Birden çok coğrafi bölgede

kullanılabilirlik sağlamaktadır. Bu da AWS'in büyük global şirketlerden biri olduğunun kanıtıdır. Sanal altyapı, çalışırken optimum kullanılabilirlik sağlayacak şekilde tasarlanmıştır. Tamamen müşteri mahremiyeti sağlamaktadır(Mathew, 2014).

Prachi vd.(2015) göre AWS ağ güvenliği, olağanüstü bir ağ güvenliğine ve olağanüstü bir ağ mimarisine sahiptir. Maliyet olarak, uygun ve kontrol edilebilirdir. Dünya standartlarında ağ mimarisinin güvenilir olduğunun düşünülmesinin nedenleri şunlardır;

- Güvenli ağ mimarisi
- Güvenli erişim noktaları
- İletim koruması
- Amazon kurumsal ayrımı
- Hata toleranslı tasarım
- Ağ izleme ve koruma

Her sunucuda olduğu gibi AWS'de de güvenlik araştırması yapılmalıdır. AWS'in güvenlik araştırma modelleri güvenilir bilgi işlem ve bilgi merkezli güvenliktir.



Şekil 9. AWS Kullanan Birimler

Yukarıdaki şekilde AWS kullanan birimler yer almıştır. AWS'i genel olarak işletmeler tercih etmektedir(Vural, 2022).

7. AWS İŞLEM HİZMETLERİ

Vural(2022) akademiye sunduğu bilgilerde AWS işlem hizmetleri için aşağıda yer alan 5 hizmeti ele almıştır.

- Amazon Elastic Compute Cloud (Amazon EC2)
- Amazon EC2 Auto Scaling
- Elastic Load Balacing
- Amazon Elastic Container Service (Amazon Ecs)
- AWS Lambda Bhargavi ve Sharma(2012) makalesinde AWS işletim hizmeti için şu bilgileri söylemiştir.

AWS platform bileşenleri şunlardır:

- Amazon Elastik Bulut Bilişimi (Amazon EC2): Bulutta ölçeklenebilirlik sağlayan bilgi işlem kapasitesidir.
- Amazon S3 (Basit Depolama Hizmeti): Basit depolama için oluşturulan web tabanlı depolama hizmetidir.
- Amazon Simple DB: Geliştiricilerin yapılandırılmış veriler üzerinde sorgu çalıştırabileceği hizmettir.
- Amazon Cloud Front: Uç konumlara dağıtım ağı hizmeti veren işlevsel platformdur.
- Amazon SQS (Basit Kuyruk Hizmetleri): Sanal makinelerde dolaşan ve depolamak için barındırılan bir hizmet sağlamaktadır.

7.1. Amazon Elastic Compute Cloud (EC2);

Programları ve web sunucuları, bu hizmetin içinde kurulabilmektedir. Bir bilgisayar makinasından farkları ise şunlardır;

- İstenilen işletim sistemi kullanılabilmektedir. (Linux, Microsoft, vb.)
- Zaman kaybı oluşturmamaktadır.
- Kurulum istenmemektedir. (Araç, makine, vb.)
- Bir yazılımı tekrar kurmayı gerektirmemektedir.
- Lokal disk kullanımını en aza düşürmektedir.
- Sistemi açınca ekrana erişim sağlamakta, kapatınca depolama havuzuna geri dönmektedir. Böylece bilgisayar hafızasında yer tutmamaktadır.
- Meta data ve user data olmak üzere iki seçenek kullanılabilmektedir(Vural, 2022).

EC2, Auto Scaling ile birlikte kullanılmaktadır. Kullanılan ram arttıkça kapasiteyi arttıran bu sistem kullanıcının isteği ve kullanımına göre kapasiteyi azaltıp, arttırabilmektedir. Buna en güzel örnek Black Friday günleridir. İnsanlar o gün içinde sitelere fazla yükleme yaptığı için bir gün önceden kapasiteyi arttırabilmektedirler. Yani günlük olarak kapasiteyi değiştirebilme özelliğine sahiptir.

Szul(2015)'a göre EC2, kullanıcıların, bir sanal özel sunucu gibi bilgisayar uygulamalarını çalıştırabilmek için sanal bilgisayarlar kiralamasına izin vermektedir. Sanal özel sunucular, işlevsellik açısından ayrılmış fiziksel sunuculara çok benzemektedir. Diğer sanal sunuculara göre daha uygun maliyetlidir. Fiziksel bir sunucu rafı satın almak, kurmak ve dağıtmak sadece dakikalarınızı almaktadır.

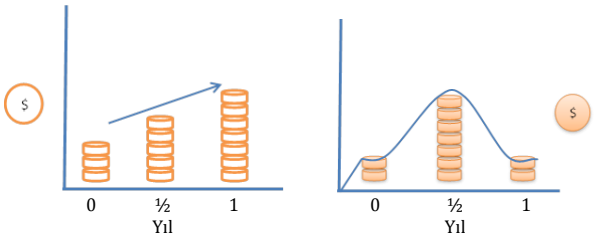
7.1.1. EC2 Kullanım örnekleri:

- Yüksek performanslı bilişim
- Büyük veri ve analiz
- Bağlı depolama
- Geliştirilmiş ağ iletişimi
- İstek üzerine bulut sunucuları
- Spot bulut sunucuları
- Rezerve edilmiş bulut sunucuları

Normal bilgisayarlarda CPU ve RAM ayrı seçilirken, EC2 birlikte seçilmektedir(Vural, 2022).

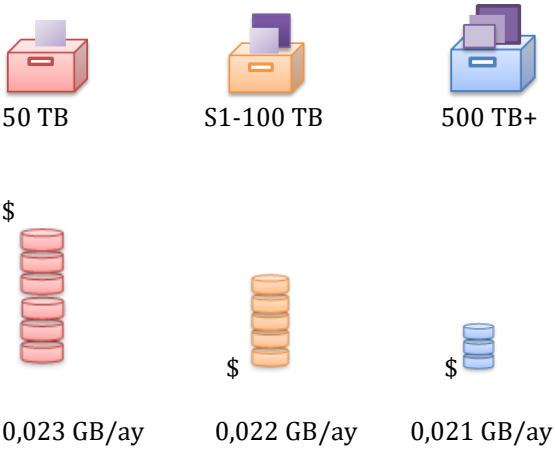
8. AMAZON EC2 ve S3 FİYATLANDIRMA

Amazonda fiyatları kullanıcılar belirlemektedir. Kullanıcının, kullanacağı yani ihtiyacı olan veri boyutu, ulaşılabilirliğine göre modelleri ve fiyatlandırmaları belirlenebilmektedir(Vural, 2022). AWS'i ücretsiz denenebileceği gibi ücretli olarak sürekli kullanıma açık olan bir platformdur. Amazon web sitesinde 3 farklı fiyatlandırma yer almaktadır. Bu 3 ayrı fiyatlandırmadan kullanıcı, kendi isteklerine özel bir depolama veya fiyatlandırma seçebilmektedir. Bunlardan ilki kullandıkça ödeme modelidir. Yalnız kullandığınız kadarını ödemenize imkân sağlayan AWS, kurumların uyumlu, çevik ve ölçeklendirme taleplerini en iyi şekilde hizmet sunmaktadır. Kullandıkça öde fiyatlandırma modeli, depolanacak verilerin değişikliklerine göre bütçeyi aşmadan, kolayca uyum sağlanmasına yardımcı olmaktadır. Bu model sayesinde işleri, tahminlere göre değil, ihtiyaçlara uygun hale getirerek kapasite tedirginliğini azaltmaktadır. İşletmelerin tam esnekliğe sahip olmasını sağlamaktadır(AWS, 2022). Amazon S3 kullanımında daha basit verileri tutmaktadır. Bu yüzden fiyatlandırmaları EC2'ya göre daha uygun olmaktadır. Örneğin Amazon S3, 100GB'lık bir depolama alanı için ortalama 2,3$ maliyet biçmektedir(Vural, 2022).



Şekil 10. AWS Amazon Kullandıkça Ödeme Modeli Karşılaştırılması (Amazon Web Services)

İkinci fiyatlandırma modeli ise, taahhütte bulunarak daha fazla tasarruf edinme modelidir. Bu modelde servisi 1 veya 3 yıllık kullanma için üye kaydı oluşturulabilmektedir. Kullanılan saat baz alınarak ödeme sağlanmaktadır. Üçüncü model ise, daha fazlasını kullanarak daha az öde modelidir. AWS'de kullanım arttıkça toplu kullanım indirimleri alarak büyük oranda tasarruf sağlayabilirsiniz.



Şekil 11. Daha Fazla Kullandıkça Daha Az Öde Modeli (Amazon Web Services)

9. AMAZON VERİ TABANI

Şekil 12. Amazon Veri Tabanları

9.1. Relational Database Service (Amazon RDS)

İlişkisel Veri Tabanı Sistemi olarak geçen RDS bir tedarik veri tabanı modelidir. Yerleşik otomatik yük devretme için Multi- Az dağıtımı yapmaktadır. Yüksek erişe bilirlik ve güvenirlik sağlamaktadır. Yoğun okuma gerektiren iş yükleri için okuma replikasının kullanılması önerilmektedir(Vural, 2022). Amazon ilişkisel veri tabanı hizmeti, bulutta bir ilişkisel veri tabanı kurmayı, çalıştırmayı ve ölçeklendirebilmeyi kolaylaştıran bir web hizmetidir. Zaman alan bir veri tabanı yönetim görevlerini yönetirken uygun maliyet sağlamaktadır. Yeniden boyutlandırılabilir kapasite sağlamaktadır. Uygulamalara ve işlere odaklanabilme konusunda kullanıcılara zaman kazandırmaktadır. Amazon RDS, bir MySQL, Oralce, SQL Server veya PostreSQL veri tabanının özelliklerine erişim sağlamaktadır. Bu mevcut veri tabanı ile birlikte hali hazırda kullanılan kod, uygulama ve araçların kullanılabileceği anlamına gelmektedir. RDS, veri tabanı yazılımına otomatik olarak yamalar uygulamaktadır ve veri tabanınızı yedeklemektedir. Yedekleri kullanıcının tanımladığı depolama süresi boyunca depolamaktadır (Mathew, 2014).

9.1.1. Amazon RDS altyapısı türleri

Altyapı türleri bellek, performans veya G/Ç için optimize edilmiştir.

- Amazon Aurora
- Postgre SQL
- MySQL
- Maria DB
- Oracle Database
- SQL Server

9.2. AMAZON DYNAMO DB

Hızlı ve esnek NoSQL anahtar değer sağlamaktadır. Aynı zamanda Dynamo DB belge veri tabanları içermektedir. Yönetim gerektiren sunucu gerektirmemektedir. Tüm ölçeklerde tek basamaklı milisaniye performans hizmeti sunmaktadır. Sürekli makine seçimi yapılmamaktadır. Kendi kendine büyüme sağlamaktadır. Maliyet ve hız avantajı sağlamaktadır. Sunucusuz web uygulamaları, mobil arka uçlar, oyun ve nesnelerin interneti kullanılırken daha çok tercih edilen hizmettir(Vural, 2022). Amazon Dynamo DB hızlı kullanım sağlamaktadır. Herhangi bir miktarda veriyi depolamayı ve almayı, her düzeyde istek trafiğine hizmet vermeyi basit ve uygun maliyetli hale getirmektedir. Tüm veri öğeleri, somut sürücülerinde (SSD) depolanmaktadır. Yüksek kullanılabilirlik ve dayanıklılık için 3 erişebilir durum alanda çoğaltılmaktadır. Dynamo DB ile düşük bir fiyat ödeyerek, yüksek oranda erişilebilir dağıtılmış bir veri tabanı kümesini çalıştırmaktadır. Dynamo DB, ölçeklendirmenin yönetimsel yükünü üzerinden atmak isteyenlerin kullanmak isteyeceği bir yöntem olduğunu düşünülmektedir. Dynamo DB, tablo için verileri ve trafiği otomatik olarak yeterli sayıda sunucuya yaymaktadır. Tutarlı ve hızlı veri muhafaza ederken etmektedir. Müşteri tarafından belirtilen talep kapasitesini ve saklanan veri miktarını yönetmek isteyen kullanıcılar için uygun görülmektedir(Mathew, 2014).

9.3. Database Migration Service(DMS)

Veri tabanlarını hızlı ve güvenli bir şekilde AWS'e geçirilmesini sağlamaktadır. Küçük sanal sunucu olarak açılım yapmaktadır. Veri tabanı aktarılması açısından en çok kullanılan veri tabanı sunucusudur.

10. SONUÇ VE TARTIŞMA

Bu araştırmada, hibrit bulut bilişim nedir, en çok kullanılan hibrit türü olan Amazon ve Amazon'un türleri araştırılmıştır. Araştırmalara göre hibrit bulut, gelişmiş ve hala gelişmekte olan bir bulut bilişim dalıdır. Hibrit bulut bilişimde yer almakta olan Amazon Web Servisi, dünya pazarının 1/3'ine sahiptir. BTK Akademi'deki bilgilere göre, Amazon'da sunucu oluşturulurken ağ bölgesi seçilmelidir. Amazon'un altyapısında sınırlı bölgeler yer almaktadır. Bu bölgeler içinde Türkiye yer almamaktadır. Türkiye'nin seçilememe nedeni ise kullanımının az olduğu için sunucu altyapısının bulunamamasıdır. Bu yüzden Türkiye'de Amazon kullanmak isteyen kişilerin ülkeye en yakın ağ olan Frankfurt seçilmektedir. Frankfurt ülkeye daha yakın olduğu için daha iyi hizmet alınabilmektedir. İncelenen makalelerde, kitaplarda ve akademide belirtilen verilere göre; Amazon'un en çok kullanılan altyapı sunucuları EC2 ve S3'dür. EC2 daha büyük ve daha işlevsel veri tutmaktadır. S3 ise daha basit yapılı verileri tutmaktadır. Amazon veri tabanı kullanımı yaparken eğer veri küçük ve öyle kalacak ise RDS veri tabanını kullanılması önerilmektedir. Kullanılacak veri alanı giderek artacak ise Dynamo DB modeli kullanılması önerilmektedir. EC2'nun daha iyi olduğunu Kaggle verilerinde yer alan "AWS EC2 Bulut Sunucusu Karşılaştırması" veri seti bu bilgiyi doğrulamaktadır. Literatürde yer alan güvenlik bilgileri dosyalarımızın kaybolmamasına yönelik bilgiler içermektedir. Fakat verilerin çalınma olasılığı hakkında açık bir bilgiye rastlanılmamıştır. Bu olasılık hakkında bir bilgiye literatür de karşılaşıldığı takdirde, bilginin güvenirliği kanıtlanamayacaktır. Günümüzde kullanılan telefonlar farkına varmadan bilgilerimizi almaktadır. Bu durum, yurt dışı kaynaklı bir bulut sunucuya ne kadar güvenebiliriz? Sorusunu akıllara getirmektedir. İnsanlar modern çağa ayak uydurmak için bulut kullanmalı mıdır? Yoksa unutulmaya başlanmış disklerle devam etmeli midir? Literatürde yer alan bilimsel çalışmalar bu kısımlara değinmemiştir. Literatürde yer alan Amazon konulu çalışmalar genel olarak Amazon'un özelliklerinden bahsetmiştir. Amazon, depolamak için disklerden daha uygun ve kaybolması %99,999999999 imkânsızdır. Verilerin kaybolmamasını isteyen ve bilgi güvenirliği konusunun çok önemli olmadığını düşünen kişiler için çok büyük nimettir. AWS verileri çok iyi bir şekilde depolayabilmektedir.

KAYNAKÇA

Narula, S., & Jain, A. (2015, February). Cloud computing security: Amazon web service. In 2015 Fifth International Conference on Advanced Computing & Communication Technologies (pp. 501-505). ieee.

Baki Onur, O. K. U. T. U. C. U. (2012). BULUT BİLİŞİM VE TEKNOLOJİLERİ (Doctoral dissertation, Okan Üniversitesi TEZ (Thesis)).

Şanlı, O. (2011). Bulut Bilişim. Akademik Bilişim, 2-4.

Amazon, E. C. (2022). Amazon ec2. línea]. Available: https://aws. amazon. com/es/ec2.

Fusaro, V. A., Patil, P., Gafni, E., Wall, D. P., & Tonellato, P. J. (2011). Biomedical cloud computing with amazon web services. PLoS computational biology, 7(8), e1002147.

Varia, J., & Mathew, S. (2014). Overview of amazon web services. Amazon Web Services, 105.

Bermudez, I., Traverso, S., Mellia, M., & Munafo, M. (2013, April). Exploring the cloud from passive measurements: The Amazon AWS case. In 2013 Proceedings IEEE INFOCOM (pp. 230-234). IEEE.

Şekil2, Snergy Research Group. https://www.srgresearch.com/ (Erişim Tarihi:25.12.2022).

Vural, Amazon Web Servisleri(AWS) ile Bulut Bilişim, BTK Akademi. Link: https://www.btkakademi.gov.tr/portal/trainer/4102 (Erişim Tarihi: 01.01.2023).

Şekil10, Amazon Web Servisi(AWS). Link: https://aws.amazon.com/tr/pricing/?nc2=h_ql_pr_ln&aws-products-pricing.sort-by=item.additionalFields.productNameLowercase&aws-products-pricing.sort-order=asc&awsf.Free%20Tier%20Type=*all&awsf.tech-category=*all (Erişim Tarihi: 29.12.2022).

Şekil11, Amazon Web Servisi(AWS). Link: https://aws.amazon.com/tr/pricing/?nc2=h_ql_pr_ln&aws-products-pricing.sort-by=item.additionalFields.productNameLowercase&aws-products-pricing.sort-order=asc&awsf.Free%20Tier%20Type=*all&awsf.tech-category=*all (Erişim Tarihi: 29.12.2022).

BİLGİSAYAR BİLİMLERİ VE TEKNOLOJİLERİ DERGİSİ
JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGIES
http://dergipark.org.tr/tr/pub/bibted

Araştırma Makalesi

Kriptografi ve Görüntü Steganografi Tabanlı Bir Veri Gizleme Uygulaması: Sten 0.1

Serhat ÇELİK[1], Nesibe YALÇIN*[1]

[1]Erciyes Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Kayseri, Türkiye

Anahtar Kelimeler:
Bilgi güvenliği
Kriptografi
Görüntü steganografi
LSB
Veri gizleme

ÖZ

Gelişen iletişim ve bilişim teknolojileri ile dijital iletişim daha hızlı ve kolay olmuş, iletişim ortamlarının kullanımı yaygınlaşmıştır. Diğer taraftan iletilen bilgiye yönelik saldırılar da sayı ve çeşit olarak artış göstermiştir. İletişim kurmak isteyen iki taraf arasında, özel ve güvenli bir iletişim ortamının sağlanmasına ihtiyaç duyulmaktadır. Bilgi güvenliği temel hedeflerinden olan gizlilik, kriptografi veya steganografi yöntemleri kullanılarak güvence altına alınabilir. Ayrıca, daha gelişmiş bir güvenlik için bu yöntemler bileştirilerek kullanılabilir. Çalışmada, veri gizliliğini sağlamaya yönelik kriptografi ve steganografi yöntemlerinin birlikte kullanıldığı bir yaklaşım önerilmiştir. Önerilen yaklaşım, görüntü dosyası içerisine veri gömme/çıkarma işlemlerini bir özet (hash) string ile gerçekleştirerek kullanıcıya ek bir güvenlik katmanı sunmaktadır. Bu işlem az bir gecikmeye neden olsa da maksimum kapasitede veri gizleme yapıldığında ulaşılan sonuçlar memnuniyet vericidir. Her bir görüntü kanalının en az anlamlı 4 bitine rastgele veri gizlenmiş görüntüler, 30 dB ve üzeri kabul edilebilir bir görüntü kalitesine sahiptir. Ayrıca, en yüksek benzerlik indeksi 0,991 olarak elde edilmiştir. Çalışma kapsamında ayrıca bir uygulama geliştirilmiştir. Geliştirilen uygulamada kriptografi yöntemleri ve en az anlamlı bit tabanlı görüntü steganografi ayrı ayrı veya birlikte çeşitli amaçlar için kullanılabilmektedir.

A Data Hiding Application based Cryptography and Image Steganography Methods: Sten 0.1

Keywords:
Information security
Cryptography
Image steganography
LSB
Data hiding

ABSTRACT

Digital communication has become easier and faster with the developing communication and information technologies, the use of communication environments has become widespread. On the other hand, attacks on transmitted information have increased in number and variety. There is a need to provide a private and secure communication environment between the two people who want to communicate securely. Confidentiality, one of information security targets, can be secured by using cryptography or steganography methods. In addition, these methods can be combined for enhanced security. This study proposes an approach that combines both methods to ensure data privacy. It provides enhanced security to the user by embedding/extracting the data in the image file with a hash. Although this process has resulted in a slight delay, the satisfaction results are obtained when random data is hiding at maximum capacity. The images with data hidden in the 4-least significant bit (LSB) of each color channel have an acceptable image quality of almost 30 dB and above. In addition, the highest similarity index is obtained as 0.991. The study also includes the development of an application. In the developed application, cryptography methods and the LSB-based image steganography can be used separately or together for various purposes.

*Sorumlu Yazar
*(nesibeyalcin@erciyes.edu.tr) ORCID ID 0000-0003-0324-9111
(serhatcelik@erciyes.edu.tr) ORCID ID 0000-0002-4717-1507
e-ISSN: 2717-8579

Geliş Tarihi: 06/06/2023l; Kabul Tarihi: 22/09/2023

1. GİRİŞ

Günümüzde bilişim teknolojilerinin hızlı gelişmesi ve yaygınlaşması ile internet kullanımı artmakta, bununla birlikte internet üzerinden gönderilen ve alınan verilerin güvenliği ciddi bir sorun olmaya devam etmektedir. Bilgi güvenliği; bilgilerin, verilerin, kaynakların, varlıkların ve değeri olan her şeyin yetkisiz erişim ve işlemlere (değiştirilme, silinme, yok edilme, izinsiz kullanma, ifşa edilme gibi) karşı korunması olarak tanımlanabilir ve üç temel başlık altında incelenebilir:

- Gizlilik; sadece yetkili kişilerin verilere erişimine izin verilmesi
- Bütünlük; verilerin yetkisiz değiştirilme veya bozulmalara karşı korunması
- Erişilebilirlik; yetkili kişilerin verilere ihtiyaç duyduklarında sürekli erişiminin sağlanması

Gizlilik, bütünlük ve erişilebilirlik ihtiyaçları, çeşitli uygulamalarda farklı şekilde vurgulanabilir. Örneğin, bir bankacılık uygulamasında kullanıcı parolasının gizli tutulması önemli iken, bireysel işlemlerin bütünlüğü korunmalı ve işlemlerin gerçekleştirilebilmesi için bankacılık sistemi erişilebilir olmalıdır. Bilgi güvenliği ihtiyaçlarını karşılamak için erişim denetimi, kimlik doğrulama, veri sınıflandırma, şifreleme, risk yönetimi, değişim kontrolü, yedekleme ve olay müdahale gibi araçlardan yararlanılmaktadır. Kriptografi ve steganografi, verileri yetkisiz kişilerden korumada en yaygın kullanılan yöntemlerdendir. Kriptografi, bir metnin içeriğini diğer bir kişinin anlamayacağı hale getirerek (şifreleyerek), steganografi ise bir veriyi başka bir verinin (görüntü, metin, ses gibi kapak/örtü dosya) içerisine saklayarak veriyi korur. Gizliliği sağlamanın en etkili yollarından biri, bu iki yöntemin birleştirilerek kullanılmasıdır.

Bilgi güvenliğini sağlamak, verileri güvence altına almak için kriptografi ve steganografiyi birleştirerek çift katmanlı bir güvenlik sunan birçok çalışma yapılmıştır. Hammad vd. (2022), steganografi ve kriptografi birleştiren bir yaklaşım benimsemiştir. Çalışmada metin önce Vigenère daha sonra Sezar şifreleme ile şifrelenmiştir. Daha sonra periyodik tabloya dönüştürülerek (her bir karakterin ASCII kodu, periyodik tablodaki bir atom numarası olacak şekilde kimyasal elementin sembolü ile değiştirilerek) şifreli metin elde edilmiştir. Elde edilen şifreli metin, örtü görüntü içerisine gömülerek gerçek bilginin varlığı gizlenmiştir. Awadh vd. (2022) tarafından veri güvenliğini sağlamak ve internet üzerinden veri alışverişi kapasitesi sorununu çözmek için hibrit bir güvenlik sistemi önerilmiştir. Önerilen sistem, bir görüntüyü başka bir görüntü içerisinde gizleme sürecini içermektedir. İlk olarak ayrık dalgacık dönüşümü kullanılarak gizlenecek görüntü sıkıştırılmış, daha sonra sıkıştırılmış görüntü Gelişmiş Şifreleme Standardı (Advanced Encryption Standard, AES) ile şifrelenmiştir. Elde edilen şifreli bitler, en az anlamlı bit (Least Significant Bit, LSB) ekleme algoritması kullanılarak başka bir görüntü içerisine yerleştirilmiş ve internet üzerinden aktarıma hazır hale getirilmiştir. AES algoritmasının kullanıldığı bir başka çalışmada (Singh ve Atria, 2015), ekstra güvenlik için gizli mesajın saklandığı görüntü dosyası şifrelenmiştir. Eliptik Eğri Kriptografisi ile görüntü steganografi birleştirilerek yapılan çalışmada (Hureib ve Gutub, 2020), gizli ve özel verilerin daha güvenli bir şekilde korunması sağlanmıştır. Adee ve Mouratidis (2022), veri güvenliğini artırmak ve bulut bilişim ortamlarında gizliliği korumak amacıyla dört aşamalı bir güvenlik modeli önermişlerdir. AES ve Rivest Shamir Adleman algoritmaları ile şifreleme ve LSB steganografi yöntemini uygulama modelin ilk iki aşamasını oluşturmaktadır. Bu aşamalardan sonra, şifreleme ve şifre çözme işlemlerinin sonuçlarına ilişkin veri yedekleme/kurtarma ve güvenli veri paylaşımı gelmektedir. Koçak tarafından yapılan çalışmada (2015), simetrik şifreleme sonucu elde edilen metin toplamda en az anlamlı 4 bit kullanılarak görüntü içerisine gizlenmiştir. Orijinal görüntü ile metnin gizlendiği görüntü benzerlik ve bozulma açısından incelenmiş, yapılan işlem sonucu görüntü kalitesinde fazla bir bozulma olmadığı belirtilmiştir. Özbilgin vd. (2018) tarafından, Vigenère şifreleme ile farklı bir şifreleme gerçekleştirilmiş ve daha sonra görüntüdeki satır, sütun veya köşegenlerde yer alan piksellerin mavi (blue, B) renk kanallarına LSB yöntemi ile gizlenmiştir. Osman vd. (2022), tek kullanımlık şerit akış şifreleme ile görüntü steganografi yöntemine dayalı hibrit bir yaklaşım önermişlerdir. Çalışmada, çeşitli görüntü formatlarında örtü görüntüleri ve farklı metin boyutları test edilmiş ve gizlenecek metin boyutunun örtü görüntüden %15 daha küçük olması gerektiği ifade edilmiştir. Ansari vd. (2020), PNG görüntülerde kullanılabilen bir şifreleme ve kümelemeye dayalı bir algoritma önermiştir. Önerilen algoritma ile PNG görüntü kullanılarak 40 bin üzeri bitin internette güvenli bir şekilde aktarımının sağlanabildiği belirtilmiştir.

Steganografi konulu çalışmalarda, LSB ekleme algoritmasının yaygın olarak ele alındığı, farklı renk kanalları ile iki veya daha fazla LSB kullanıldığı da görülmüştür. Şahin vd. (2006), geliştirdikleri kullanıcı arayüzü aracılığı ile farklı boyutlardaki 24-bit renkli bitmap görüntü dosyaları üzerinde LSB yöntemini kullanarak metin gizleme ve çıkarma yapmışlardır. Koçak çalışmasında (2015), kırmızı (red, R) ve yeşil (green, G) renk kanalları ile her bir kanaldaki en az anlamlı 2 biti gizli metni gömmek için kullanmıştır. Doğan vd. (2016) ise görüntülerdeki 0. ve 2. bitler veri gömme için tercih etmişlerdir. Hureib ve Gutub tarafından yapılan çalışmada (2020), olumlu ve olumsuz yönlerini ortaya koyabilmek amacıyla bir ve iki LSB ekleme test edilmiştir. Mesajın bir pikselin R, G ve B

kanallarının rastgele bit konumlarına gömülmesi yaklaşımının önerildiği bir çalışmada (Ali vd., 2019) standart LSB ekleme yöntemi ile aynı PSNR değerleri elde edilmiştir. Bununla birlikte mesaj pikselin bitlerine ardışık yerleştirilmediği için ek bir güvenlik sunmaktadır. LSB ekleme yöntemine bit kaydırma işleminin eklendiği çalışmada (Solak ve Altınışık, 2019), anahtar ile metin önce şifrelenmiş daha sonra şifrelenmiş metinde kaydırma yapılarak gizlenecek metin elde edilmiştir. 8 bitlik bir veri için RGBBGRRG kanalları kullanılmış ve gizlenecek metin sırasıyla bu kanalların en az anlamlı bitlerine yerleştirilmiştir. Yakut (2022a) tarafından LSB yöntemine dayalı önerilen yaklaşımda, gizlenecek mesaj taşıyıcı verinin en az anlamlı bitlerinde herhangi bir değişiklik olmaksızın iletilmektedir. İletilen bitler ve taşıyıcının bitleri XOR işlemine tabi tutulur ve sonuç verisi üretilir. Bu şekilde gizlenecek mesajın tespiti mümkün olmamaktadır, ancak sonuç verisinin gönderimi alıcıya ek bir yük getirmektedir.

Balkesen ve Koçer tarafından yapılan çalışmada (2020), AES ile şifrelenmiş metni görüntülere gizlemek için rastgele piksel seçim yaklaşımı benimsenmiştir. Söz konusu yaklaşımda seçilen piksel bilgisi, görüntüde gizlenmekte ve böylece gizlenen metnin doğru çıkarılması sağlanmaktadır. Rastgele piksellere bit yerleştirmenin yapıldığı bir diğer çalışmada (Emam vd., 2016), bir karakterin yerleştirilmesi 1. piksel BG kanalları ve 2-LSB, 2. piksel B kanalı ve 1-LSB, 3. piksel BG kanalları ve 2-LSB ekleme şeklinde gerçekleştirilmiştir. Bhardwaj ve Sharma (2016) tarafından rastgele seçilmiş piksellere ters bit LSB yöntemi kullanılarak 8 bitlik gri tonlamalı bir görüntü seti üzerinde veri gizleme deneyleri yürütülmüştür. Önerilen yöntem ile veri gizlenmiş görüntülerin, standart LSB yöntemi ile elde edilenlerden daha yüksek görsel kaliteye sahip olduğu belirtilmiştir. Bu çalışmaların en önemli kısımlarından biri rastgele sayı üretimidir. Yakut (2021, 2022b) çalışmalarında bu konu üzerine odaklanmış ve ayrık kosinüs dönüşümünü kullanan bir rastgele sayı üreteci önermiştir. Gri tonlamalı görüntüler içerisine gizlenmiş mesajın tespitini zorlaştırmak için önerilen bir yöntemde (Macit ve Koyun, 2020) ise örtü görüntü ilk olarak bloklara ayrılmış ve her bir blok için çeşitli hesaplamalar yapılara görüntüye ilişkin daha az ayrıntı içeren bloklar tespit edilmiş ve gizli mesaj yerleştirilmiştir. Böylece görüntünün tamamı yerine bazı bloklarına veri gizlendiğinden standart LSB çıkarma yöntemi, gömülü metnin eldesi için kullanılamayacaktır. Çalışma sonucunda blok sayısını artırmanın, veri kapasitesini artırdığı diğer taraftan karmaşıklığı ve işlem süresini de artırdığı ifade edilmiştir. Baysan ve Özekes (2023) tarafından veri gizleme için uzaklaştırılmış LSB ekleme yöntemi önerilmiştir. Önerilen yöntem ile gizlenecek mesaj, görüntü içerisine orantılı olarak dağıtılarak belirli bir alanda yığılma engellenmiştir. Konyar vd. (2018) tarafından ise kullanıcıların veri gizleme için en uygun görüntüyü seçebilmelerine olanak tanıyan bir arayüz tasarlanmıştır. Gizleme kapasitesi için RGB kanallarının hepsi kullanılmış ve gizleme sonrası en az değişikliğe uğrayan görüntünün kullanıcıya önerilmesi sağlanmıştır.

Bu çalışmada, bazı klasik kriptografi yöntemleri ve LSB görüntü steganografi algoritması kullanılarak ve özet string aracılığı ile verileri gizleyerek üç katmanlı bir güvenlik uygulaması "Sten 0.1" geliştirilmiştir. Uygulama ile çeşitli formattaki görüntü dosyaları üzerinde farklı renk kanalları ve LSB sayısı seçimi ile veri (orijinal ya da şifrelenmiş metin) gizleme işlemi mümkündür. Çalışmada kullanılan kriptografi ve steganografi yöntemleri Bölüm 2'de açıklanmıştır. Geliştirilen uygulamaya ilişkin detaylar Bölüm 3'te, uygulama sonuçları ise Bölüm 4'te verilmiştir. Son bölümde, sonuçlar değerlendirilmiş ve öneriler sunulmuştur.

2. BİLGİ GÜVENLİĞİ SİSTEMLERİ

Bilişim teknolojisi alanındaki gelişmelerle birlikte, verilerin güvenliğini sağlamak için birçok yöntem uygulanmaktadır. Kriptografi ve steganografi, veri gizliliğini sağlamada kullanılabilecek yöntemlerdir. Bu bölümde, klasik kriptografi yöntemleri ve görüntü steganografi yöntemi olan LSB ekleme sunulmuştur.

2.1. Kriptografi

Kriptografi, bir mesajı yetkili olmayan kişiler için anlaşılması zor bir forma dönüştürerek koruma altına almayı amaçlar. Mesajın yalnızca alıcı ve gönderici tarafından okunmasına, görüntülenmesine izin verir ve üçüncü şahıslara, kötü niyetli taraflara karşı haberleşmenin güvenliğini sağlar. Geliştirilmiş birçok kriptografi algoritması vardır ve anahtar kullanımı, anahtar yapısı/türü, karmaşıklık ve performans açısından farklılık göstermektedir. Çalışma kapsamında yerine koyma, yer değiştirme, tek alfabeli, çok alfabeli, blok şifreleme gibi farklı özellikler sunan klasik kriptografi yöntemleri tercih edilmiştir. Bu yöntemlerin ortak özelliği ise şifreleme ve şifre çözme için sadece bir gizli anahtar kullanmasıdır.

- Sezar şifreleme; tek alfabeli bir yerine koyma şifreleme yöntemidir. Romalı lider Julius Caesar tarafından ordu haberleşmesinde mesajların güvenliğini sağlamak için kullanılmıştır (Abraham ve Shefiu, 2012). Şifrelenecek metindeki her bir harf yerine, o harften anahtar olarak belirlenen miktar kadar ileri kaydırma (öteleme) yapıldığında karşılık gelen harf kullanılarak şifreleme yapılır. Şifre çözme (deşifre etme) işlemi için ise yapılan işlemin tersi uygulanır.
- Scytale (sarmal) şifreleme; en bilinen yer değiştirme şifrelemesidir. Scytale olarak adlandırılan bir tahta çubuğun etrafına sarılmış parşömen veya papirüs şeridine gizlemek istenilen mesaj yazılır. Şerit açıldığında orijinal mesajdaki tüm harfler farklı bir konuma aktarılmış ve böylece şifrelenmiş metin elde edilmiş olur (Diepenbroek,

2021). Mesajın deşifre edilebilmesi için aynı yarıçapa sahip Scytale cihazının kullanılması gerekir.

- Vigenère şifreleme; çoklu alfabe kullanan bir yerine koyma şifreleme türüdür. Alfabedeki harfler satır ve sütun isimlerini ifade etmek üzere bir tablo oluşturulur (Aliyu ve Olaniyan, 2016). Tablonun ilk satırına orijinal alfabe yazılır, daha sonra satırdaki harfler birer kaydırılarak bir sonraki satıra yazılacak alfabe elde edilir ve bu işlem tablonun bütün satırları için tekrarlanır. Şifreleme işleminde oluşturulan bu tabloda satır, şifrelenecek metnin harflerini; sütun ise anahtar metnin/kelimenin harflerini temsil eder. Seçilen anahtar metnin boyutu, şifrelenecek metinden daha kısa uzunlukta ise anahtar metin, şifrelenecek metni tamamen örtene kadar yinelenir. Sırasıyla şifrelenecek metnin her bir harfi ile anahtar metnin aynı sıradaki (indeksteki) harfine tabloda karşılık gelen harf yazılarak şifreli metin oluşturulur.
- Hill şifreleme; bir blok şifreleme türüdür ve lineer cebire dayanır. Şifrelenecek metin belirli bir büyüklükteki bloklara ayrılarak bloklar halinde şifrelenir. Blok boyutuna (k) göre şifrelemede kullanılacak anahtar kare matrisin boyutu (k × k) belirlenir. Şifreleme işleminde kullanılan anahtar matrisin elemanlarını (ASCII kod için 0-255 aralığında) belirlemek önemli bir sorundur. Şifre çözme işlemi de düşünüldüğünde matrisin bir tersi olmalıdır ve anahtarın determinantı 1 olacak şekilde elemanlar seçilmelidir (Putera vd., 2016). Şifrelenecek metindeki harflerin alfabedeki sırasından ya da ASCII değerlerinden oluşan her bir blok, anahtar matris ile çarpılır ve daha sonra kullanılan alfabedeki harf sayısına ya da 256'ya göre modu alınır ve elde edilen sayısal değerlere karşılık gelen harfler ile şifreli metin elde edilir.

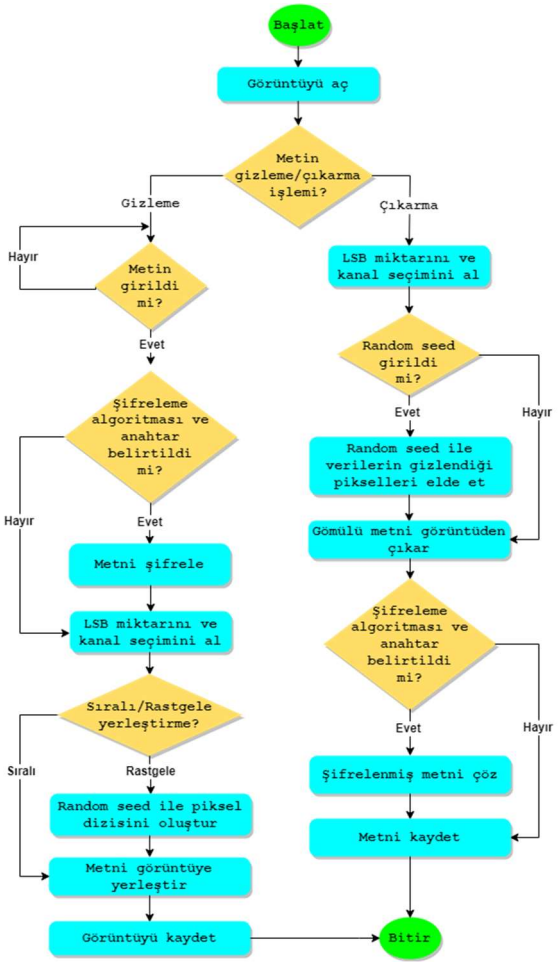
2.2. Steganografi

Steganografi, gizlilik, mahremiyet, veri bütünlüğü gibi bilgi güvenliği kavramlarını sağlamak için yaygın olarak kullanılır. "Gizlenmiş yazı" anlamına gelir ve veriyi yetkisiz kişilerden gizleme amacını taşır. Veriyi gizlemek için görüntü, metin, video gibi çeşitli ortamlar kullanılır, bu ortama örtü (cover, kapak) ve içerisinde gizlenmiş veri bulunan ortama ise stego denir. Elde edilen tüm stego ortamlar için kalite, çok önemli bir parametredir. Objektif kalite değerlendirmesi yapmak temel doğruluk için gereklidir.

Bu çalışmada, LSB tabanlı görüntü steganografi yöntemi kullanılmıştır. LSB ekleme, sıklıkla uygulanan bir steganografi yöntemidir ve daha az karmaşıktır. Gizlenecek metin, ikili sayı (binary) sisteminde ifade edilir ve daha sonra görüntünün her bir pikselinin en az anlamlı bitlerine sırasıyla yerleştirilir. Yerleştirmede LSB kullanımı, orijinal görüntü üzerinde gözle görülür bir değişiklik oluşturmamaktadır. RGB görüntülerde her bir renk kanalının en az anlamlı bitleri kullanılarak da veri gizlenebilir. Ayrıca farklı kanal(lar) ve/veya LSB miktarı tercih edilerek güvenlik artırılabilir. En uygun gizleme yöntemini belirlemek ve örtü görüntü ile stego görüntü arasındaki farkı değerlendirmek için Tepe Sinyal Gürültü Oranı (Peak Signal-to-Noise Ratio, PSNR) ve Ortalama Karesel Hata (Mean Squared Error, MSE) kalite ölçümleri yaygın olarak kullanılmaktadır (Sara vd., 2019).

3. UYGULAMA

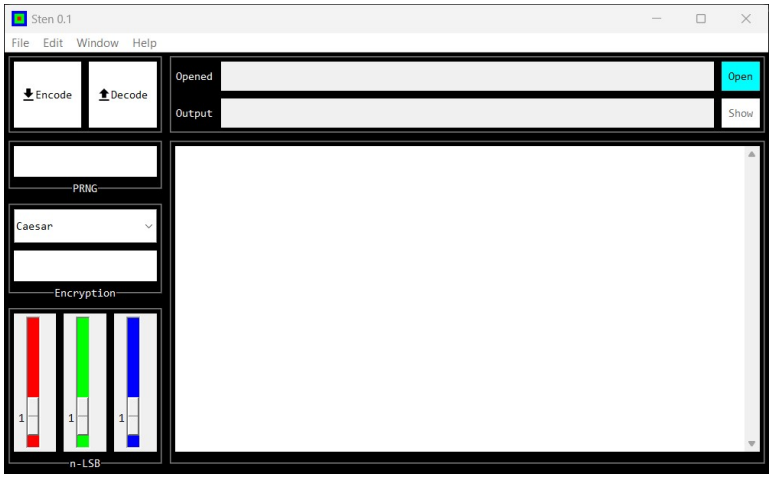
Çalışmada, çeşitli klasik kriptografi yöntemlerini ve LSB görüntü steganografiyi birlikte kullanan Sten 0.1 uygulaması geliştirilmiştir. Uygulama aracılığıyla metin şifreleme ve şifre çözme, bir metnin orijinal veya şifrelenmiş halini bir örtü görüntüye gizleme ve çıkarma yapılabilmektedir. Süreci özetleyen akış diyagramı Şekil 1'de verilmiştir.



Şekil 1. Sten 0.1 uygulaması işlem akışı

3.1. Sten 0.1

Sten 0.1 uygulaması, Python dilinde Tkinter Grafiksel Kullanıcı Arayüzü (Graphical User Interface, GUI) kütüphanesi yardımıyla geliştirilmiştir. Uygulama ile yapılabilecek şifreleme, şifre çözme, veri yerleştirme/gömme ve çıkarma gibi işlemleri son kullanıcıya en etkili biçimde sunabilmek amacıyla Şekil 2'de verilen menüye dayalı GUI tasarlanmıştır.



Şekil 2. Sten 0.1 kullanıcı arayüzü

3.2. Şifreleme ve şifre çözme

Arayüz ile kriptografi için tercih edilen şifreleme yöntemi (Sezar, Scytale, Vigenère ve Hill şifreleme) seçilebilmekte ve seçilen yönteme ait bir gizli anahtar girişi yapılabilmektedir (bkz. Şekil 3). Uygun anahtar seçiminin yapılabilmesi için çeşitli kontroller gerçekleştirilmektedir. Örneğin, Sezar şifrelemede sadece sayı girişine izin verilmektedir. Bu durumun dışında kalan her giriş reddedilmektedir. Aynı şekilde, "Decode" işleminin yapılabilmesi için de bu bilgilerin girilmesi gerekmektedir. Şekil 3'te Sezar şifrelemeye ait girilen şifreleme anahtarı *** olarak gözükmektedir. Bu, ekstra bir güvenlik gerekçesiyle yapılmıştır. Kullanıcı isterse uygulamaya ait konfigürasyon (yapılandırma) dosyasında değişiklik yaparak bu davranışı değiştirebilir. Bu şekilde girilen anahtar değeri varsayılan olarak görünür olacaktır.

Şekil 3. Şifreleme algoritması seçimi ve anahtar girişi

Şifreleme işlemlerinin gerçekleştirilmesinde ele alınan hususlar aşağıda listelenmiştir.

- Sezar şifreleme için girilen anahtar değerinin (öteleme miktarı) sayı olup olmadığı kontrol edilir ve alfabe uzunluğunun katlarına eşit olmadığından emin olunur.
- Scytale şifreleme ve şifre çözme için aynı özellikte Scytale cihazı kullanılmalıdır. Bu nedenle cihazın yarıçap değeri, uygulamada anahtar olarak kullanılmıştır. Şifrelemede metin matris (satır × sütun) formatında ele alındığında anahtar değeri, sütun sayısına (sonraki satıra geçme adımına) karşılık gelir. Şifrelenecek metnin harfleri, satır ve sütunlara sırasıyla yerleştirilir ve sonra tek boyutlu diziye dönüştürülerek şifrelenmiş metin elde edilir. Anahtar değerinin sıfırdan farklı bir sayı olarak girilmesi sağlanır.
- Vigenère ve Hill şifrelemede, girilen anahtar metnin karakterlerinin alfabede olduğundan emin olunur.
- Hill şifreleme işleminde seçilen bir metin, önceden belirlenmiş bir kare matrise metnin her harfinin ASCII tablosundaki sayısal karşılığı yerleştirilerek anahtar oluşturulur. Matriste eksik kalan kısımlar rastgele sayılarla tamamlanır. Ancak her metin, anahtar olarak kullanılamaz. Metnin anahtar olarak kullanılabilmesi için anahtar ile oluşturulacak matrisin tersinin alınabiliyor olması ve aynı zamanda bu matrisin determinantının, kullanılacak alfabedeki karakter sayısıyla aralarında asal olması gerekmektedir. Bunun sebebi, şifre çözme işleminde matrisin ters çevrilecek olmasıdır. Eğer matrisin determinantı alfabe uzunluğu ile aralarında asal olmazsa ortak bölenleri 1'den farklı gelir. Bu, anahtar olarak kullanılamaz. Bu şartları sağlayan bir anahtar seçildikten sonra, şifrelenecek metin anahtar matrisin bir boyutunun uzunluğunda bloklara ayrılır. Bloklar ile anahtar matris çarpılır, elde edilen sayıların ASCII karşılıkları bulunur ve şifrelenmiş metnin blokları elde edilir. Şifre çözme işlemi ise anahtar matrisin tersi ile şifrelenmiş metnin her bir bloğunun matris çarpımına sokulması ile gerçekleştirilir.

3.3. Veri gizleme ve çıkarma

3.3.1. Örtü görüntüsü seçimi

Örtü görüntü, kullanıcıdan alınan ve içerisine gizli mesajın yerleştirilmesi amacıyla kullanılan görüntü dosyasıdır. LSB yöntemi, yapısı gereği sıkıştırmasız görüntü formatlarıyla uğraşmayı gerektirir. Bu yüzden çalışmada JPEG tarzı sıkıştırılmış görüntü formatları yerine RGB veya RGBA renk modunda BMP ve PNG görüntülerle çalışılmıştır. Uygulamada Şekil 4'te gösterilen 24 bit derinliğine sahip dört farklı test örtü görüntüsü üzerinde deneyler gerçekleştirilmiştir. "Lena" ve "Pepper" standart görüntü dosyaları 512×512, "Cat" görüntüsü (URL-1) 900×900 ve "Scenery" (URL-2) ise 1280×853 piksel boyutundadır.

Seçilen örtü görüntüye ait faydalı bilgiler "File→Image Properties" seçeneği ile (boyut, renk modu, veri kapasitesi gibi) görüntülenebilmektedir. Veri gizleme öncesi görüntünün veri yükleme kapasitesini görmesi açısından kullanıcıya avantaj sağlamaktadır.

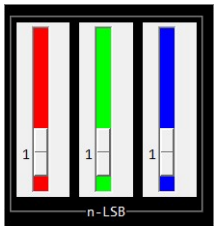
a)  b) 

c)  d) 

Şekil 4. Test görüntüleri, a) Lena, b) Pepper, c) Cat ve d) Scenery

3.3.2. Verinin örtü görüntüye sıralı yerleştirilmesi

Gizlenecek metnin ve örtü görüntünün boyutuna bağlı olarak kullanıcı, arayüz ile kanal(lar) ve LSB miktarı tercihi yapılabilmektedir. Bu tercih, gizli verinin örtü verisine yerleştirilmesi ve stego verisinden gizli verinin çıkarılması aşamasında bir anahtar görevi görecektir. Verilerin gizlenmesinde önceki çalışmalarda kullanılan, bir, iki veya ikiden fazla LSB kullanımına alternatif olarak kullanıcıya, renkli bir görüntünün her bir renk kanalı için ayrı ayrı LSB miktarını belirleme seçeneği sunulmuştur. RGB renk modunda bir görüntü için minimum 1 ve maksimum 24 bit kullanılabilir. Kullanıcı arayüzü üzerinde her bir kanal için LSB miktarının belirlenebilmesini sağlayan kısım Şekil 5'te gösterilmiştir.



Şekil 5. Test görüntüleri, a) Lena, b) Pepper, c) Cat ve d) Scenery

Her bir piksel için n-LSB kullanımı durumunda, görüntü içerisine yerleştirilebilecek maksimum bit sayısı (kapasite), Denklem (1) yardımıyla hesaplanmaktadır. 1-LSB kullanımı durumunda elde edilen değerler, Lena görüntüsü için 786.432, Cat görüntüsü için 2.430.000 ve Scenery görüntüsü için 3.275.520 bittir.

kapasite = n × piksel genişlik
× piksel yükseklik
× renk kanalı sayısı (1)

Gizlenmek istenen veri, gizleme seçenekleri ile birlikte "gizleme fonksiyonuna" verilir. Bu fonksiyon gizli veriyi örtü görüntüye orijinal veya şifrelenmiş hali ile yerleştirerek stego görüntünün elde edilmesini sağlar. Ardından stego görüntünün kaydedileceği konum bilgisi alınır ve işlemin başarılı olması durumunda bir bilgilendirme mesajı kullanıcıya gösterilir.

3.3.3. Verinin örtü görüntüye rastgele yerleştirilmesi

Ek opsiyonel bir güvenlik adımı olarak gizlenecek veri bitleri doğrudan örtü görüntüye gömülmek yerine rastgele seçilmiş piksellerine yerleştirilebilmektedir. Bunun için bir Sözde Rastgele Sayı Üreteci (Pseudo-Random Number Generator, PRNG) seed (tohum) değeri, anahtar görevi üstlenecek şekilde kullanıcıya sunulmuştur (bkz. Şekil 2). PRNG, bir seed değeri alan ve bir dizi benzersiz sayı üreten bir kara kutu görevi görmektedir (Emam vd., 2016). Uygulama kapsamında, kullanıcıdan bir string (dizge) değer alınır. Girilen değer için PRNG yardımı ile sabit uzunlukta bir özet string elde edilir. Anahtar değer girilmeden de renk kanalı ve LSB miktarı seçimi girdi olarak da alınabilmektedir. Böylece ek bir anahtar değere ihtiyaç ortadan kalkmaktadır. Son olarak özet kullanılarak görüntüdeki piksellerin indislerinin rastgele dağılmış bir versiyonu elde edilir. Böylece kullanılan veri bitleri de saklanarak gömülü metnin standart LSB çıkarma yöntemi ile tespit edilme şansı azaltılmaktadır. Eğer bu işlem tercih edilmek istenmez ise veriler görüntü içerisine sıralı olarak yerleştirilecektir. İlgili işleme ait sözde kod aşağıda verilmiştir.

Girişler : Gizlenecek mesaj, *M*
Şifreleme algoritması, *C*
Şifreleme anahtarı, *K*
PRNG, *R*
Resim dosyası pikselleri, *P*
R, G ve B için LSB, $L = \{LSB_R, LSB_G, LSB_B\}$
Çıkış : Stego görüntü, *S*

```
if M is None then
    return None
else
    if C is not None then
        if K is None then
            return None
        else
            sifrelemeAlgoritmasiKullan = True
        end
    else
        sifrelemeAlgoritmasiKullan = False
    end
end
if R is not None then
    P piksellerini listeye dönüştür
    P piksellerini R değerine göre rastgele karıştır
else
    P piksellerini listeye dönüştür
end
if sifrelemeAlgoritmasiKullan is True then
    M mesajını C şifreleme algoritması ve K şifreleme anahtarı ile şifrele
    L değeri ile M mesajını P piksellerine yerleştirerek S stego nesnesini oluştur
else
    L değeri ile M mesajını P piksellerine yerleştirerek S stego nesnesini oluştur
end
return S
```

3.3.4. Gizli verinin stego görüntüden çıkarılması

Stego görüntü, arayüz aracılığıyla seçildikten sonra şifreleme ve veri gizleme işlemlerinde kullanılan şifreleme yöntemi ve anahtarı, renk kanalı ve LSB miktarı seçimi ile PRNG seed uygulanma durumu için aynı bilgiler girilir. Bu bilgiler, "uygulama tarafından gizlenmiş veriyi tekrar elde etme fonksiyonuna" gönderilir. Bu şekilde fonksiyon önceden yapılan işlemleri tersten yürüterek orijinal metnin elde edilmesini sağlar.

4. SONUÇLAR VE TARTIŞMA

Uygulama kapsamında Şekil 4'te verilen örtü görüntülere çeşitli alternatiflerle aynı boyutta veri gizlenmiş işlem süreleri incelenmiştir. Sıralı ve rastgele seçilmiş görüntü piksellerine farklı LSB miktarında yerleştirme yapılarak elde edilmiş stego görüntüler, örtü görüntülerle MSE, PSNR ve Yapısal Benzerlik İndeksi Ölçümü (Structural Similarity Index Measurement, SSIM) bakımından karşılaştırılmış ve görüntülerdeki bozulma oranları analiz edilmiştir.

MSE, orijinal görüntü (O) ile stego görüntü (O') arasındaki hatanın karesel ortalamasını verir ve Denklem (2) yardımı ile hesaplanır (Balkesen ve Koçer, 2020). Görüntülerdeki pikseller, h×w boyutunda bir matris olarak ele alınır.

$$MSE = \frac{\sum_{i=1}^{h}\sum_{j=1}^{w}\left(O_{(i,j)} - O'_{(i,j)}\right)^2}{h \times w} \quad (2)$$

MSE, Denklem (3) ile PSNR'ye dönüştürülür. Orijinal görüntü ile stego görüntü arasındaki farkı değerlendirmek için bir kalite ölçümü olarak kullanılır. Desibel (dB) cinsinden ölçülür (Ali ve ark., 2019; Ansari ve ark., 2020). Daha yüksek bir PSNR, PSNR'nin daha iyi olduğunu gösterir (Awadh ve ark., 2022). X, orijinal görüntünün tepe sinyal değeridir ve Denklem (4) kullanılarak hesaplanır (Bhardwaj ve Sharma, 2016).

$$PSNR = 10 \times \log_{10}\frac{X^2}{MSE} \quad (3)$$

$$X = max\left(O_{(i,j)} - O'_{(i,j)}\right) \quad (4)$$

SSIM, iki görüntünün ne kadar benzer olduğunu belirlemek için kullanılır ve değeri ne kadar yüksekse o kadar iyidir. Denklem (5) yardımı ile hesaplanır. c_1 ve c_2, her terimi stabilize eden küçük pozitif sabitlerdir. μ_O ve σ_O, orijinal görüntüye ilişkin piksel ortalamasını ve standart sapmasını temsil etmektedir. $\mu_{O'}$ ve $\sigma_{O'}$ ise sırasıyla stego görüntüye ilişkin piksel ortalaması ve standart sapmasıdır (Awadh ve ark., 2022).

$$SSIM(O, O') = \frac{2 \times \mu_O \times \mu_{O'} + c_1}{\mu_O^2 + \mu_{O'}^2 + c_1} \times \frac{2 \times \sigma_{OO'} + c_2}{\sigma_O^2 + \sigma_{O'}^2 + c_2} \quad (5)$$

Piksellerin R, G ve B kanalları için ayrı ayrı 1-LSB, 3-LSB ve 4-LSB ekleme sıralı ve rastgele şekilde uygulanmış ve görüntü içerisine maksimum bit yerleştirilmesi yapılmıştır. 1-LSB, 3-LSB ve 4-LSB yöntemleri sonucu hesaplanan PSNR, MSE ve SSIM değerleri ve stego görüntünün oluşturulması için harcanan süreler, Tablo 1, Tablo 2 ve Tablo 3'te sırasıyla karşılaştırmalı olarak verilmiştir. Deneyler sonucu elde edilen görüntüler ise Tablo 4'te sunulmuştur.

Tablo 1. RGB kanalları için 1-LSB kullanıldığında elde edilen deneysel sonuçlar

Görüntü	Maksimum Kapasite (bit)	Gizlenen Karakter Boyutu (bayt)	PSNR (dB)		MSE		SSIM		Gizleme Süresi (sn)	
			Sıralı	Rastgele	Sıralı	Rastgele	Sıralı	Rastgele	Sıralı	Rastgele
Lena	786.432	98.304	51,141	51,143	0,319	0,303	0,997	0,997	1,321	1,341
Pepper	786.432	98.304	51,138	51,133	0,313	0,302	0,996	0,997	1,320	1,411
Cat	2.430.000	303.750	51,071	51,069	0,484	0,486	0,998	0,998	2,994	3,360
Scenery	3.275.520	409.440	51,143	51,146	0,317	0,306	0,997	0,997	4,767	5,356

Tablo 2. RGB kanalları için 3-LSB kullanıldığında elde edilen deneysel sonuçlar

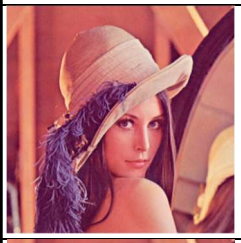
Görüntü	Maksimum Kapasite (bit)	Gizlenen Karakter Boyutu (bayt)	PSNR (dB)		MSE		SSIM		Gizleme Süresi (sn)	
			Sıralı	Rastgele	Sıralı	Rastgele	Sıralı	Rastgele	Sıralı	Rastgele
Lena	2.359.296	294.912	38,395	38,394	2,083	2,102	0,994	0,994	1,323	1,412
Pepper	2.359.296	294.912	38,285	38,285	2,395	2,426	0,990	0,990	1,320	1,411
Cat	7.290.000	911.250	38,833	38,838	9,133	9,031	0,995	0,995	3,001	3,655
Scenery	9.826.560	1.228.320	38,327	38,313	2,210	2,214	0,990	0,990	5,096	5,723

Tablo 3. RGB kanalları için 4-LSB kullanıldığında elde edilen deneysel sonuçlar

Görüntü	Maksimum Kapasite (bit)	Gizlenen Karakter Boyutu (bayt)	PSNR (dB)		MSE		SSIM		Gizleme Süresi (sn)	
			Sıralı	Rastgele	Sıralı	Rastgele	Sıralı	Rastgele	Sıralı	Rastgele
Lena	3.145.728	393.216	32,691	32,678	13,373	13,358	0,991	0,991	1,326	1,419
Pepper	3.145.728	393.216	32,861	32,851	8,955	9,088	0,981	0,981	1,320	1,411

Cat	9.720.000	1.215.000	29,217	29,216	67,967	67,973	0,992	0,992	3,317	3,742
Scenery	13.102.080	1.637.760	32,803	32,807	13,008	13,003	0,979	0,979	5,182	5,955

Tablo 4. Sıralı ve rastgele 1-LSB, 3-LSB ve 4-LSB yerleştirme sonucu elde edilen görüntüler

Yerleştirme	n-LSB	Stego Görüntü			
		Lena	Pepper	Cat	Scenery
Sıralı	1-LSB				
	3-LSB				
	4-LSB				
Rastgele	1-LSB				
	3-LSB				
	4-LSB				

Sıralı piksellere ve rastgele seçilmiş piksellere 1-LSB (bir pikselin her bir kanalına 1 bit olmak üzere toplamda 3 bit) ekleme yönteminin uygulanması elde edilen stego görüntülerin benzerlik indeksleri, %99,6 ve üzeridir. Bununla birlikte her iki durumda da birbirine yakın PSNR değerleri elde edilmiş ve 0,002 - 0,005 dB arasında bir değişim gözlenmiştir (bkz. Tablo 1). MSE değerleri açısından ise rastgele 1-LSB yerleştirme sonucu Cat görüntüsü hariç diğer görüntülerde azalma görülmüştür. Diğer taraftan 0,002 değerinde hata artışına sebep olarak, Cat görüntüsünün baskın bir arka planının olması gösterilebilir.

En yüksek boyuta sahip Scenery görüntüsüne, maksimum kapasitede, rastgele piksellere 4-LSB ekleme işlemi uygulandığında elde edilen stego-görüntü kalitesi (PSNR değeri 32,803 dB), sıralı ekleme sonucu elde edilen stego-görüntü kalitesiyle

(PSNR değeri 32,807 dB) hemen hemen aynıdır. Bu sonuçlar, önerilen rastgele bit yerleştirme yönteminin etkinliğini ortaya koymaktadır.

1-LSB kullanılması durumu ile karşılaştırıldığında, 4-LSB eklenmiş görüntülerdeki değişim insan gözünün algılayabileceği boyutlara ulaşabilir. Tablo 4'te sunulan görüntüler incelendiğinde, özellikle beyaz bir arka plana sahip olan Cat görüntüsü için sıralı yerleştirme sonucu elde edilen görüntüdeki farklılık çıplak gözle tespit mümkün olabilmektedir. Belirgin özellikleri olan görüntüler için çalışmada önerilen rastgele yerleştirme yöntemi tercih edilebilir.

İşlem sırasında ekstra hesaplamalar nedeniyle veri gizleme için harcanan süre, LSB miktarı ve gizlenen veri boyutu ile genel olarak artmıştır. Sıralı 4-LSB yerleştirme ile karşılaştırıldığında rastgele 4-LSB yerleştirme ile veri gizleme sürelerinde Lena görüntüsü için 0,07; Pepper için 0,069; Cat için 0,128 ve Scenery için 0,149 oranında artış görülmüştür. Diğer taraftan, verilerin görüntü içerisinde rastgele piksellerin en az anlamlı bitlerine atanması daha yüksek güvenlik seviyesi sunmaktadır.

Sıralı ve rastgele yerleştirme işlemleri sonucunda yaklaşık %98 ve üzeri SSIM değerine sahip stego görüntüler elde edilmiştir. Kullanılan LSB miktarının ve gizlenen veri boyutunun artması ile elde edilen PSNR değerlerinde bir düşüş gözlense de stego görüntüler, kabul edilebilir bir görsel kaliteye (30 dB ve üzeri) sahiptir. Ayrıca rastgele yerleştirme ile ihmal edilebilir derecede bir gürültü sağlanmıştır. Elde edilen deneysel sonuçlar, literatürdeki çalışmalar ile de karşılaştırılmış ve Tablo 5'te sunulmuştur.

Tablo 5. Diğer araştırmacılar tarafından elde edilen sonuçlar ile çalışmanın karşılaştırılması

Referans	Yöntem	Görüntü	Gizlenen Veri (bit)	PSNR (dB)	SSIM
(Koçak, 2019)	RG ve sıralı 2-LSB	Lena	1.046.519	29,885	0,986
		Pepper	1.046.519	27,963	0,984
(Ansari vd., 2020)	1-LSB	Lena	15.160	68,45	-
		Pepper	15.160	67,09	-
	3-LSB	Lena	15.160	51,89	-
		Pepper	15.160	51,89	-
	4-LSB	Lena	15.160	46,28	-
		Pepper	15.160	46,28	-
(Doğan vd., 2016)	Sıralı 2-LSB	Lena	12.282	34,34	0,997
(Solak ve Altınışık, 2019)	RGBBGRRG ve 1-LSB	Lena	698.984	51,656	0,99982
		Pepper	698.984	51,621	0,99981
(Balkesen ve Koçer, 2020)	RGB ve rastgele 1-LSB	Lena	693.600	51,29	0,98
(Emam vd., 2016)	3 piksel, BG ve rastgele 2-1-2 LSB	Lena	349.520	51,83	-
		Pepper	349.520	51,85	-
Bu çalışma	RGB ve rastgele 1-LSB	Lena	786.432	51,143	0,997
		Pepper	786.432	51,133	0,997
	RGB ve rastgele 3-LSB	Lena	2.359.296	38,394	0,994
		Pepper	2.359.296	38,285	0,990
	RGB ve rastgele 4-LSB	Lena	3.145.728	32,678	0,991
		Pepper	3.145.728	32,851	0,981

Koçak (2015) tarafından Lena ve Pepper görüntüleri üzerinde yapılan testlerde, R ve G kanallarına 2-LSB (bir piksel için toplam 4 bit) ekleme ile 1.046.519 bit yerleştirme yapılmıştır. Belirtilen görüntüler için sırasıyla PSNR değerleri 29,885 dB - 27,963 dB ve SSIM değerleri 0,986 ve 0,984 olarak elde edilmiştir. Çalışma kapsamında R, G ve B kanalları için 3-LSB (bir piksele toplamda 9 bit) ekleme yöntemi ile 2.359.296 bit yerleştirilmiş ve PSNR değeri en düşük 51,133 dB ve SSIM değeri 0,997 olarak hesaplanmıştır. Önerilen yöntem ile daha yüksek PSNR ve SSIM değerlerinin bulunması, görüntü üzerinde daha az bozulmaya neden olunduğunu gösterir.

Renk kanallarının her birine 1-LSB yöntemini uygulayan ve toplamda 3 piksele 8 bit yerleştirme yapan Solak ve Altınışık (2019), 698.984 bit veri gizlenmiş stego Lena ve Pepper görüntüleri için PSNR değerlerini sırasıyla 51,656 dB ve 51,621 dB olarak bulmuşlardır. Bu çalışmada, her bir renk kanalına rastgele 1-LSB (toplamda 3 piksele 9 bit) yerleştirme deneyleri sonucunda elde edilen PSNR değerleri ise sırasıyla 51,143 dB ve 51,133 dB'dir. Çalışma kapsamında, (Solak ve Altınışık, 2019) ile oldukça yakın PSNR ve SSIM değerleri elde edilmekle birlikte, daha fazla veri gizlenmiş (786.432 bit) ve bitler rastgele yerleştirildiği için daha fazla güvenlik sunulmuştur. Lena görüntüsünün RGB kanallarına rastgele 1-LSB ekleme işleminin yapıldığı çalışmada

(Balkesen ve Koçer, 2020), 693.600 bit veri gizlenmiş ve 51,29 dB PSNR değeri elde edilmiştir. Rastgele bit yerleştirmenin yapıldığı diğer çalışmada (Emam vd., 2016) ise 349.520 bit veri gizlenmiş ve PSNR değeri 51,83 dB hesaplanmıştır. Önerilen yöntemde, aynı görüntünün RGB kanallarına 1-LSB ekleme ile daha fazla veri gizlenmiş ve buna rağmen PSNR değeri 51,143 dB elde edilmiştir. SSIM değerleri açısından karşılaştırıldığında 0,997 ile görüntü için daha yüksek yapısal benzerlik sağlanmıştır.

5. DEĞERLENDİRME

Çalışmada, hassas verileri gizlemek ve saldırılardan korumak için kriptografi (gizli bilgileri şifreleme) ve steganografi (gizli bilgileri saklama) yöntemlerini kullanan bir güvenlik sistemi önerilmiş ve Sten 0.1 isimli bir uygulama geliştirilmiştir. Uygulama ile kriptografi ve steganografi işlemleri kullanıcı seçimine bağlı olarak ayrı ayrı gerçekleştirilebilmektedir. Ayrıca, LSB ekleme işleminin bir rastgele dizi dikkate alınarak yapılması ile veri gizleme daha güvenli hale getirilmiştir. Çalışmada önerilen yöntemde, rastgele karıştırma sürecinden dolayı çıktının üretilmesi biraz daha yavaş olabilmektedir. Ancak gizlenen metnin elde edilmesi zorlaştırılmış ve ek bir anahtar kullanımı ile ekstra güvenlik sunulmuştur.

Kullanılan LSB miktarı arttıkça normal veya rastgele yerleştirme fark etmeksizin stego görüntüde gözlenen farklılık gözle tespit edilebilir boyutta olacaktır. Özellikle Şekil 4b'deki görüntü gibi baskın renklerin bulunduğu durumlarda bu vaziyet daha da ciddidir. Rastgele yerleştirme yöntemi, özellikle belirgin arka plana sahip görüntüler ve/veya daha fazla LSB kullanılması gereken durumlar için artan güvenlik sunduğundan önerilmektedir. Bununla birlikte normal yerleştirme sonucu üretilen çıktı üzerinde benzer bir etki oluşturduğundan düşük veri gizleme süre artışı göz ardı edilerek artan güvenlik sebebiyle tercih edilebilir.

Çalışma kapsamında geliştirilen Sten 0.1 uygulaması, tıbbi kayıtlar, ihale verileri gibi önemli verilerin saklanması, korunması ve güvenli iletimi amaçlı kullanılabilir. Uygulamada kullanılan klasik şifreleme yöntemlerine modern yöntemler de eklenerek kullanıcılara daha gelişmiş bir güvenlik sunulabilir.

KAYNAKÇA

Abraham, O., ve Shefiu, G. O. (2012). An Improved Caesar Cipher (ICC) Algorithm. *IJESAT International Journal of Engineering Science & Advanced Technology*, 2(5), 1198-1202.

Adee, R., ve Mouratidis, H. (2022). A Dynamic Four-Step Data Security Model for Data in Cloud Computing Based on Cryptography and Steganography. *Sensors*, 22(3), 1109.

Ali, U. A. M. E., Sohrawordi, M., & Uddin, M. P. (2019). A Robust and Secured Image Steganography Using LSB and Random Bit Substitution. *American Journal of Engineering Research (AJER)*, 8(2), 39-44.

Aliyu, A. A. M., ve Olaniyan, A. (2016). Vigenère Cipher: Trends, Review and Possible Modifications. *International Journal of Computer Applications*, 135(11), 46-50.

Ansari, A., Mohammadi, M. S., & Ahmed, S. S. (2020). Digital colour image steganography for PNG format and secured based on encoding and clustering. *International Journal of Engineering Research and Technology*, 13(2), 345-354.

Awadh, W. A., Alasady, A. S., & Hamoud, A. K. (2022). Hybrid Information Security System via Combination of Compression, Cryptography, and Image Steganography. *International Journal of Electrical and Computer Engineering*, 12(6), 6574-6584.

Balkesen, C., ve Koçer,H. E. (2020). Embedding Encrypted Data into an Image with a Random Pixel Layout Approach. *European Journal of Science and Technology*, (Special Issue), 123-130.

Baysan, B., ve Özekes, S. (2023). DLSB - Uzaklaştırılmış En Önemsiz Bit Steganografi. *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, 38 (3), 1725-1736.

Bhardwaj, R., ve Sharma, V. (2016). Image Steganography Based on Complemented Message and Inverted bit LSB Substitution. *Procedia Computer Science,* 93, 832-838.

Diepenbroek, M. (2021). Secret Communication in Antiquity the Spartan Scytale. *Ancient Warfare* XIV-3, 44-47.

Doğan, F., Dağ, R., & Türkoğlu, İ. (2016). İmgeler İçin Farklı Bir Veri Gizleme Yaklaşımı. *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi*, 7(3), 501-514.

Emam, M. M., Aly, A. A., & Omara, F. A. (2016). An Improved Image Steganography Method based on LSB Technique with Random Pixel Selection. *International Journal of Advanced Computer Science and Applications*, 7(3), 361-366.

Hammad, R., Latif, K. A., & Amrullah, A. Z. vd. (2022) Implementation of Combined Steganography and Cryptography Vigenère Cipher, Caesar Cipher and Converting Periodic Tables for Securing Secret Message. *Journal of Physics*, 2279(1), 012006(1-6).

Hureib, E. S., ve Gutub, A. A. (2020). Enhancing Medical Data Security via Combining Elliptic Curve Cryptography with 1-LSB and 2-LSB Image Steganography. *IJCSNS International Journal of Computer Science and Network Security*, 20(12), 232-241.

Koçak, C. (2015). Kriptografi ve Stenografi Yöntemlerini Birlikte Kullanarak Yüksek Güvenlikli Veri Gizleme. *Erciyes Üniversitesi Fen Bilimleri Enstitüsü Fen Bilimleri Dergisi*, 31(2), 115-123.

Konyar, M. Z., İlkin, S., Çelik, N., & Sondaş, A. (2018). Steganografi için En Uygun Resmi Belirleyen Uygulama Arayüz Tasarımı. *İleri Teknoloji Bilimleri Dergisi*, 7(1), 83-89.

Macit, H. B., ve Koyun, A. (2020). A New Imperceptible Steganography Method for Grayscale Images. *Mühendislik Bilimleri ve Tasarım Dergisi*, 8 (2), 357-365.

Osman, O. M., Kanona, M. E. A., Hassan, M. K., Elkhair, A. A. E., & Mohamed, K. S. (2022). Hybrid Multistage Framework for Data Manipulation by Combining Cryptography and Steganography. *Bulletin of Electrical Engineering and Informatics*, 11(1), 327-335.

Özbilgin, F., Durmuş, F., & Karagöl, S. (2018). Yazılı Metni Şifreleyip LSB Yöntemi ile Gizleme. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, 6(3) (Özel Sayı: UMAS 2017), 676-685.

Putera, A., Siahaan, U., & Rahim, R. (2016). Dynamic key matrix of hill cipher using genetic algorithm. *International Journal of Security and Its Applications*, 10(8), 173-180.

Sara, U., Akter, M., & Uddin, M. (2019). Image Quality Assessment through FSIM, SSIM, MSE and PSNR—A Comparative Study. *Journal of Computer and Communications*, 7, 8-18.

Singh, S., ve Atria, V. K. (2015). Dual Layer Security of Data Using LSB Image Steganography Method and AES Encryption Algorithm. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 8(5), 259-266.

Solak, S., ve Altınışık, U. (2019). A New Approach for Steganography: Bit Shifting Operation of Encrypted Data in LSB (SED-LSB). *Bilişim Teknolojileri Dergisi*, 12(1), 75-81.

Şahin, A., Buluş, E., & Sakallı, M. T. (2006). 24-Bit Renkli Resimler Üzerinde En Önemsiz Bite Ekleme Yöntemini Kullanarak Bilgi Gizleme. *Trakya Üniversitesi Fen Bilimleri Dergisi*, 7(1), 17-22.

Yakut, S. (2021). Random Number Generator Based on Discrete Cosine Transform Based Lossy Picture Compression. *Naturengs*, 2(2), 76-85.

Yakut, S. (2022a). Steganography Approach Based on the Least Significant Bit Technique. 6th International Artificial Intelligence & Data Processing Symposium, 8-9 Sep. 2022, Malatya, Türkiye, 165-169.

Yakut, S. (2022b). Kayıplı Resim Sıkıştırma Algoritmalarını Temel Alan Rastgele Sayı Üreteci. *Adıyaman Üniversitesi Mühendislik Bilimleri Dergisi*, 9(18), 571-580.

URL-1: https://i.pinimg.com/originals/f9/25/e1/f925e13343ffc8726316f519b3619424.png [Erişim Tarihi: 11.05.2023]

URL-2: https://demo.joomlalabs.com/images/slideshow/1280x853/stars-345902_1280.jpg [Erişim Tarihi: 11.05.2023]

BİLGİSAYAR BİLİMLERİ VE TEKNOLOJİLERİ DERGİSİ
JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGIES
http://dergipark.org.tr/tr/pub/bibted

Araştırma Makalesi

DtyPAM: Kurumsal Destek Firmaları için Önerilmiş Konteynır Tabanlı Ayrıcalıklı Erişim Yönetim Sistemi

Hamza Kürşat ŞİMŞEK*[1], Halil ARSLAN[1], Yasin GÖRMEZ[2]

[1]Sivas Cumhuriyet Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği, Sivas, Türkiye
[2]Sivas Cumhuriyet Üniversitesi, İktisadi ve İdari Bilimler Fakültesi, Yönetim Bilişim Sistemleri, Sivas, Türkiye

Anahtar Kelimeler:
Ayrıcalıklı erişim yönetim,
Uzaktan destek,
Uzaktan çalışma,
Kod olarak altyapı,
Mikroservis

ÖZ

Bilişim alanında önceki zamanlarda da uygulanan uzaktan destek ve uzaktan çalışma kavramları, 2019 yılında başlayan ve tüm dünyayı etkisi altına alan COVID-19 salgını ile hemen hemen tüm sektörler tarafından uygulanmaya başlamıştır. Ölçeği ne olursa olsun bütün girişimler dijital uygulamaları kullanmakta ya da kullanma planı yapmaktadır. Özellikle holding düzeyindeki firmalar, birçok iş sürecini karmaşık kurumsal kaynak planlama uygulamaları üzerinden yürütmektedir. Bu uygulamalar içinse genellikle dış kaynaklardan destek almakta ve bu destekler günümüzde sıklıkla uzaktan yapılmaktadır. Bu aşamada kurumlar güçlü bir erişim yönetim sistemine ihtiyaç duymaktadırlar. Bahsedilen sebeplerden ötürü çalışmamızda, uzaktan bağlantı ve destek süreçlerinin sanal masaüstü ve kod olarak alt yapı teknolojileri kullanılarak otomatik şekilde yapılabileceği bir ayrıcalıklı erişim yönetim sistemi önerilmiştir. Tasarlanan sistem ile kullanıcılara, bağlantı sağlanacak sunucuda yapılacak olan iş için en az düzeyde ayrıcalık verilmesi hedeflenmektedir. Bir sunucuya yapılan bağlantıların geriye dönük takibinin rahatlıkla yapılabilmesi için, çalışma sonucu önerilmiş olan ayrıcalıklı erişim yönetim uygulamasına güçlü bir kayıt defteri sistemi (log) eklenmiştir. Çalışmamızda konteynır teknolojileri kullanılarak mikro-servis tabanlı bir sistem önerilmiş bu sayede platform bağımsız çalışan bir sistem elde edilmesi amaçlanmıştır. Çalışmamız sonucunda elde edilen sistemin kod olarak altyapı ve konteynır teknolojilerini birlikte kullanan ilk ayrıcalıklı erişim yönetim sistemi olduğu değerlendirilmektedir.

DtyPAM: Container Based Privilege Access Management System for Corporate Consulting Companies

Keywords:
Privilege Access management,
Remote support,
Remote work,
Infrastructure as code,
Microservice

ABSTRACT

The concepts of remote support and remote work, which have been applied in the field of information technology in the past, began to be implemented by almost all sectors with the onset of the COVID-19 pandemic in 2019, affecting the entire world. Regardless of scale, all enterprises are using digital applications or planning to use them. Especially at the level of holding companies, many business processes are conducted through complex enterprise resource planning applications. These applications often require support from external sources, and such support is frequently provided remotely in today's world. In this context, organizations need a robust access management system. For the reasons mentioned above, in our study, we propose a privileged access management system that can automatically perform remote connection and support processes using virtual desktop and code as infrastructure technologies. The designed system aims to provide the minimum level of privilege to users for the tasks to be carried out on the connected server. To facilitate the retrospective tracking of connections to a server, a strong registry system (log) has been added to the privileged access management application recommended as a result of the study. In our study, a microservices-based system using container technologies is proposed, aiming to achieve a platform-independent system. The system obtained as a result of our study is considered to be the first privileged access management system that uses both code infrastructure and container technologies.

***Sorumlu Yazar**

*(h.kursatsimsek@gmail.com) ORCID ID 0009-0009-5274-8698
(harslan@cumhuriyet.edu.tr) ORCID ID 0000-0003-3286-5159
(yasingormez@cumhuriyet.edu.tr) ORCID ID 0000-0001-8276-2030

e-ISSN: 2717-8579
Geliş Tarihi: 12/07/2023; Kabul Tarihi: 01/12/2023

1. GİRİŞ

Ayrıcalıklı Erişim Yönetimi (Privilege Access Management - PAM), ayrıcalıklı kullanıcıların bir dizi güvenlik işlevi ve süreci aracılığıyla sistemlere veya kaynaklara nasıl eriştiklerini kontrol eden, yöneten ve raporlayan bir kimlik güvenliği çözümüdür. PAM, herhangi bir sisteme erişimi denetlemek için kullanılabilir, ancak genellikle etki alanı denetleyicileri ve üretim sunucuları gibi yüksek değerli kaynaklara daha sık uygulanır (Garbis & Chapman, 2021).

PAM, herhangi bir kuruluşun güvenlik stratejisinin temel bir bileşenidir. Bu sistemler, yalnızca yetkili kullanıcıların belirli görevleri yerine getirebilmesini ve hassas verilere erişebilmesini sağlayarak, kuruluş içindeki ayrıcalıklı hesapların ve erişim düzeylerinin kullanımını kontrol etmek ve izlemek için tasarlanmıştır. Bu amaç doğrultusunda iyi bir PAM çözümü iki faktörlü kimlik doğrulama, belirli eylemler için onay talep etme ve tüm etkinlikler için bir denetim izi sağlama gibi süreçleri de barındırmalıdır. PAM çözümünün en önemli faydalarından biri, sistemlere ve verilere yetkisiz erişimi engellemeye yardımcı olmasıdır. Kurum çalışanları, farklı sistemlere bağlantı için bağımsız üyelikler kullanabilmektedir. Bu durumda işten ayrılan bir personel üyeliklerinin iptal edilmesi karmaşık bir hal almakta, gözden kaçan durumlarda ise kötü niyetli personellerin ihlaller yapabilmesine zemin oluşturulmaktadır. PAM çözümleri üyelik sistemlerini tek bir noktada toplayarak, işten çıkan personellerin tüm sistemlere ve verilere erişiminin hızlı ve kolay bir şekilde iptal edilmesini sağlamaktadır. Bu yapısı sayesinde çalışanların neden olduğu veri ihlallerine ve diğer güvenlik olaylarına karşı korunmaya yardımcı olur. Ayrıcalıklı bir erişim yönetim sisteminin diğer bir yararı ise endüstri düzenlemelerine ve standartlarına uygunluğun sağlanmasına yardımcı olmasıdır. Örneğin, sağlık veya finans gibi denetime tabii sektörlerde, hassas verilere erişimde katı kontroller istenebilmektedir. PAM çözümü işletmeler için ekstra bir güvenlik katmanı sağlarken, hassas veriler için kontrollerin yapılabilmesine yardımcı olmaktadır. PAM, harici saldırganlar ve kazara veri ihlalleri gibi çeşitli tehditlere karşı koruma sağladığı için siber güvenliğin en önemli yönlerinden biri olarak değerlendirilmektedir (Garbis & Chapman, 2021). Tüm bu faydaların yanı sıra iyi bir PAM çözümü;

- Hassas verilere ve kritik sistemlere erişimi yalnızca yetkili kullanıcılarla sınırlayarak, bu bilgilere yetkisiz erişimi ve kötüye kullanımı önlemeye yardımcı olur.
- İçeriden gelen tehditler, genellikle bir kuruluşun sistemlerine ve verilerine yasal erişime sahip oldukları için özellikle zarar verici olabilir. PAM, kullanıcıların erişimini, görevlerini yerine getirebilmek için gereken minimumla sınırlayarak ve çok faktörlü kimlik doğrulama gibi kontroller uygulayarak içeriden gelen tehdit riskini azaltmaya yardımcı olur.
- Dış saldırganlar, oturum açma kimlik bilgilerini çalarak veya tahmin ederek bir kuruluşun sistemlerine ve verilerine erişmeye çalışabilir. Ayrıcalıklı erişim yönetimi, güçlü parolalar ve parola yönetimi çözümleri uygulayarak bu saldırıların önlenmesine yardımcı olur.
- Kazara veri ihlalleri, yetkili kullanıcılar hata yaptığında veya ayrıcalıklarını kasıtlı olarak kötüye kullandığında meydana gelebilir. Ayrıcalıklı erişim yönetimi, kullanıcıların erişimini yalnızca ihtiyaç duydukları kaynaklarla sınırlayarak ve etkinlik izleme gibi denetimler uygulayarak bu ihlallerin etkisini azaltmaya yardımcı olur.

Bu çalışmada PAM çözümlerinin tüm bu özellikleri dikkate alınarak, kurumsal destek sistemlerine entegre olabilen modern ve yerli ayrıcalıklı erişim sistemi içeren, host bağımsız masaüstü altyapısı üzerinden oturumları yönetebilen, kod olarak altyapı (Infrastructure as Code - IaC) ile kullanıcı kurulumlarına gerek kalmadan hazır sistem sunabilen, olası bağlantı sayısının limit sorununu çözebilen, güçlü bir aksiyon kayıt sistemi barındıran, kimlik bilgileri güvenliği sağlayarak yönetim süreçlerinin kolaylaştıran ve sürekli erişilebilir olan bir PAM sistemi önerilmiştir. Önerilen sistemin geliştirilmesi sürecinde konteynır teknolojileri kullanılarak mikro-servis mimarisi kullanılmıştır. Sistemin ana amaçlarından birisi, uzak sunuculara erişim için güvenlik ve ayrıcalık sağlamaktır. Uzak sunuculara erişim için ssh ve rdp gibi farklı ağ protokolleri kullanılmaktadır. Çalışma ile yapılacak uygulamada yerinde ve bulut üzerinde konumlandırılabilen, bilgi güvenliği ihmallerine karşı dirençli, farklı ağ protokolleri ile uyumlu ve ileride çıkabilecek yeni ağ protokollerine uyum sağlayabilecek bir PAM sisteminin tasarlanması amaçlanmaktadır. Sistemin mikro-servis tabanlı geliştirileceği için her bir ağ protokolü için farklı bir mikro-servis geliştirilecektir. Uzaktan erişim için yeni bir ağ protokolünün gerekli olması durumunda sadece o ağ protokolü için bir mikro-servis geliştirilecek, bu sayede sistemin tümünde değişiklik yapmadan yeni ağ protokolleri sisteme eklenebilecektir. Tasarlanan sistem ile kullanıcılara, bağlantı sağlanacak sunucuda yapılacak olan iş için en az düzeyde ayrıcalık verilmesi hedeflenmektedir. Bir sunucuya yapılan bağlantıların geriye dönük takibinin rahatlıkla yapılabilmesi için, çalışma ile tasarlanacak olan PAM uygulamasına güçlü bir kayıt defteri sistemi (log) eklenmesi düşünülmektedir. Eklenecek olan bu kayıt sisteminin, veri madenciliği ve iş zekâsı gibi analizlere de uyumlu olması, platformun ölçeklenebilmesi ve sürekli çalışabilmesi ise çalışmamızın diğer amaçları arasında yer almaktadır.

1.1. Literatür araştırması

PAM sistemleri, bir kuruluş içindeki ayrıcalıklı kullanıcıların kritik kaynaklara erişimini güvence altına almak için tasarlanmıştır. PAM çözümleri, kullanıcı erişimi üzerinde ayrıntılı kontrol sağlar, kullanıcı etkinliğini izler ve en az ayrıcalık ilkelerini uygular. Kuruluşlar, iş süreçlerini yürütmek için dijital varlıkları gün geçtikçe daha sık kullanmaktadırlar. Bu doğrultuda ayrıcalıklı hesaplara ve kaynaklara erişimi güvence altına almak, firmaların güvenliğini ve teknolojik değişime uyumunu sürmesi için büyük önem arz etmektedir. Bu bağlamda literatürde PAM çözümleri için birçok çalışma yapılmıştır. Tep ve diğerleri bulut sistem atakları hakkında bir literatür araştırması yaparak temel güncel saldırılar ve bu saldırıları azaltma üzerine önerilerde bulunmuştur. Yaptıkları literatür araştırmasında elde ettikleri bilgileri kullanarak bir PAM çözümü önermiş ve önerdikleri sistemi irdelemişlerdir (Tep, Martini, Hunt, & Choo, 2015). Sindiren ve Ciylan çalışmalarında şirket içi atakları analiz ederek bu tür saldırıların azaltılması için önerilerde bulunmuşlardır. Bunun yanı sıra PAM çözümlerine katkı sağlamak için en az ayrıcalık verme, iş dağılımları ve sosyal mühendislik için personel bilinçlendirme süreçleri için prosedürler geliştirmişlerdir (Sindiren & Ciylan, 2018). Steinhoff yaptığı çalışma ile PAM limitlerini ve gereksinimlerini irdelemiş ve geliştirilen PAM sistemleri için konteynır teknolojilerinin kullanımının getireceği faydaları analiz etmiştir (Steinhoff, 2020). Tabrizchi ve diğerleri çalışmasında bulut bilişim bileşenlerinin güvenlik ve gizlilik açısından analizini yapmış, bu bağlamda karşılaşılan sorunları analiz etmiş ve bu sorunlara çözüm yolları önermişlerdir. Yapılan çalışma sonucu elde edilen çıktıların bulut üzerinde çalışan PAM sistemleri için ciddi fayda sağlayacağı ön görülmektedir (Tabrizchi & Kuchaki Rafsanjani, 2020). Purba yaptığı çalışma ile kuruluşların, kritik bilgi teknolojilerini varlıklarını korumak, uyumluluk düzenlemesini karşılamak ve veri ihlallerini önlemek için PAM kullanmasının önemini vurgulamış ve yayında kuruluşların ISO 27001 kontrolünü karşılayan PAM çözümü elde etmesi için önerilerde bulunmuştur (Anton & Soetomo, 2018). Sindiren ve Ciylan yaptıkları çalışmada imtiyazlı hesapları minimum maliyetle kontrol edebilmek, yönetebilmek ve takip edilebilmek için bir model tasarlamışlardır. Bu uygulama modeli, ayrıcalıklı kullanıcı hesaplarının parolalarının temel bilgi teknolojisi güvenlik ilkelerine uygun olarak belirlenmesinde ve daha güçlü parolaların oluşturulmasını katkı sağlamaktadır (Sindiren & Ciylan, 2019). Ylonen ve diğerleri yaptıkları çalışmada kuruluşların SSH kullanıcı anahtarlarının yönetimine odaklanarak bir kuruluşta SSH etkileşimli ve otomatikleştirilmiş erişim yönetiminin temelleri hakkında bilgi vermişlerdir (Ylonen, Turner, Scarfone, & Souppaya, 2015). D'Silva ve diğerleri yapmış oldukları çalışmada çevrimiçi korumaya ilişkin Sıfır Güven (Zero Trust) ilkesinin başarısını analiz etmişlerdir. Sıfır Güven uygulaması ve araştırması ile ilgili literatürü tarayarak gelecekteki ağ güvenliği için Sıfır Güven ilkesini irdelemiştir. Çeşitli saldırı türlerine yanıt veren mimariyi uygulamak için konteynır teknolojisini kullanmıştır. Açık sistem ara bağlantısı (Open Systems Interconnection) modelinin her katmanında Sıfır Güven Mimarisinin avantaj ve dezavantajlarına odaklanmıştır. Çalışma ile önerilmiş olan sistemin PAM alt yapılarında kullanılan sistemler için ciddi faydaları olacağı ön görülmektedir (D'Silva & Ambawade, 2021). Xu ve diğerleri yapmış oldukları çalışmada konteynır teknolojisini kullanarak dağıtılmış rol tabanlı erişim denetimi tabanlı bir kontrol mekanizması önermişlerdir. Konteynır tabanlı erişim kontrol mekanizmasının avantaj ve dezavantajlarını tartışarak yetki devri nedeniyle oluşabilecek tehlikeleri çözmek için önerilerde bulunmuşlardır (Lang, Jiang, Ding, & Bai, 2019). Alruwies ve diğerleri günümüz dünyasında sistemlere erişim yetki kontrolünün gerekliliğinden ve zorluklarından bahsettikleri çalışmada Active Directory tabanlı bir PAM sistemi önermişlerdir (Alruwies, Mishra, Abdul, & Alshehri, 2021). Ionita, PrivX isimli PAM uygulamasında çalışabilen bir veri tabanı güvenlik eklentisi geliştirmiştir. Geliştirmiş olduğu eklenti sonrasında yapay veri ile yapmış olduğu performans analizi sonucunda gecikme medyan değerinin 304 milisaniye olduğunu hesaplamış ve sağlanan güvenlikten dolayı bu gecikmenin kabul edilebilir olduğunu değerlendirmiştir (Ionita, 2023). Tran yapmış olduğu çalışmada bir sistemden başka bir sisteme erişim için kullanılacak PAM uygulamasının sistem bilgileri tanımlama, sistem protokolleri tanımlama, kimlik bilgileri kurulum sürecini belirleme ve kimlik bilgileri güncelleme olmak dört prosedürünü oluşturmuştur (Tran, 2020). Preuveneers ve Joosen PAM sistemlerini irdeledikleri çalışmalarında açık kaynak kodlu birleşik kimlik ve erişim yönetimi çözümü olan OpenAM kullanılarak sağlık hizmetleri için yeni bir prosedür oluşturmuşlardır (Preuveneers & Joosen, t.y.).

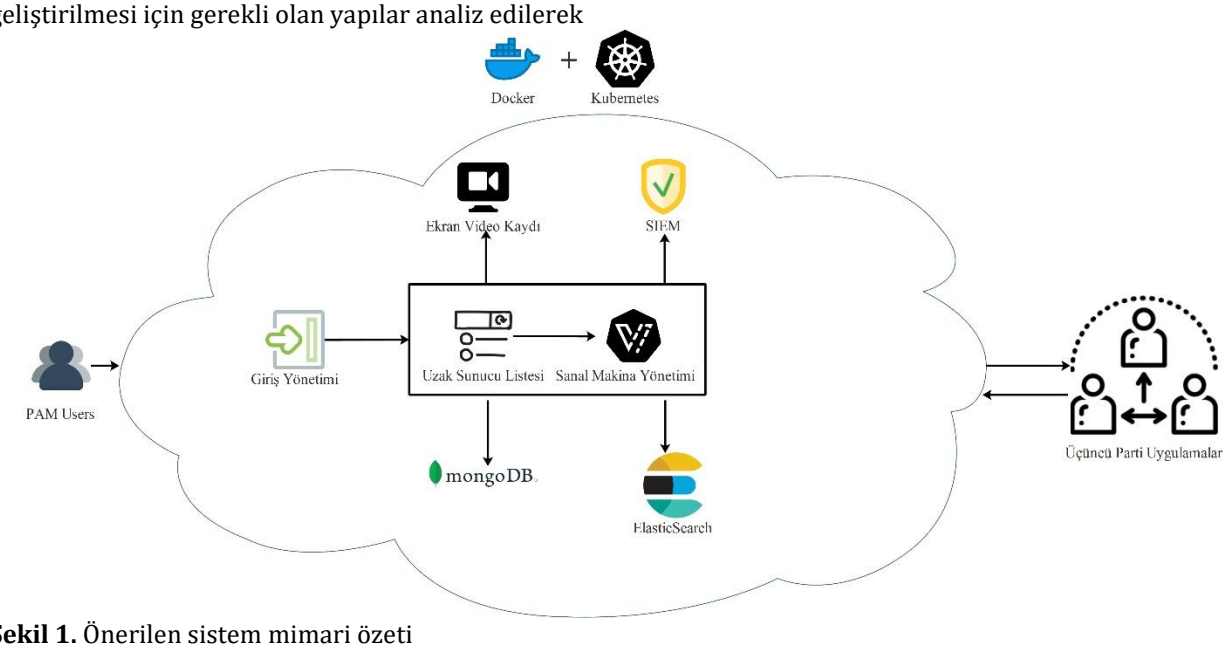
Literatürde var olan çalışmalar incelendiğinde ayrıcalıklı erişim yönetim sistemlerinin modern siber güvenlik stratejilerinin kritik bir bileşeni olduğu kanısına varılmıştır. PAM sistemlerinin, ayrıcalıklı kullanıcıların kritik dijital kaynaklara erişimini kontrol etmek ve izlemek için tasarlandığı ve hassas verilere veya sistemlere yalnızca yetkili kullanıcıların erişebildiğini sağladığı gözlemlenmiştir. Güvenlik gereksinimlerini operasyonel verimlilikle dengelerken, karmaşık ve heterojen ortamlarda çalışabilen kapsamlı ve uyarlanabilir PAM çözümlerinin önemli olduğu literatür araştırması sonucunda anlaşılmıştır.

Literatürde var olan çalışmalar incelenerek tasarlanmış olan önerilen sistemde, benzer çözümlerde bulunan özelliklerin bulunmasına dikkat edilmiştir. Bunun yanı sıra literatürde var

olan sistemlerin hiç birinde kod olarak alt yapı teknolojisi kullanılarak otomatik erişim kolaylığı sağlama özelliklerinin bulunmadığı görülmüştür. Önerilen sisteminin özgün yanı ise, uzaktan destek süreçlerinde kullanılmak üzere sanal makine tabanlı sistemlerin kod olarak teknolojisi ile konfigüre edilmesi olarak değerlendirilmektedir. Bu yapısı ile çalışmamızın, PAM araştırmalarına yeni bir vizyon kazandıracağı ön görülmektedir.

2. YÖNTEM

Çalışmamızda önerilen PAM sisteminin geliştirilmesi için gerekli olan yapılar analiz edilerek belirlenmiştir. Bu kapsamda sistemimizi geliştirmek için konteynır teknolojisi, konteynır orkestrasyon aracı, kod olarak altyapı geliştirme aracı, dokuman tabanlı veri tabanı, tam metin indeksleme aracı ve sanal makine konfigürasyon teknolojisine ihtiyaç duyulduğu kanaatine varılmıştır. Teknoloji dünyası araştırılarak bu isterileri karşılayacak açık kaynak kodlu yazılımlardan Docker, Kubernetes, IaC mimarisi, MongoDB, ElasticSearch ve Kubevirt çalışmamızda kullanılmak üzere tercih edilmiştir. Önerilen yöntemin mimari yapısı Şekil 1 ile gösterilmektedir.



Şekil 1. Önerilen sistem mimari özeti

2.1. Docker

Aynı işletim sistemi üzerinde birbirinden izole birden fazla işletim sisteminin çalıştırılabilmesine sanallaştırma denir. Sanallaştırma ile birlikte son zamanlarda konteynır yapısı da teknoloji dünyasında sıkça kullanılmaya başlanmıştır. Sanallaştırma teknolojilerinden farklı olarak daha küçük boyutlara ve bir fonksiyonu çalıştırmak için minimum sistem gereksinimlerine sahip olması, konteynır teknolojisinin sanallaştırmaya alternatif olarak kullanılmasının önemli sebeplerindendir. Önerilen sistemin taşınabilir, diğer sistemlerden yalıtılmış ve buluta dağıtılabilir olması için konteynır yapısından faydalanılmış ve bu kapsamda Docker teknolojisi kullanılmıştır ("Docker Documentation", 200M.S.).

2.2. Kubernetes

Konteynır teknolojilerinin kullanım sıklığının artması ile birlikte özellikle yatay ölçekleme yapabilmek, sistemi sürekli ayakta tutabilmek ve sistem sağlığını kontrol edebilmek için konteynır orkestrasyon araçları kullanılmaya başlanmıştır. Konteynır teknolojileri önermiş olduğumuz sistemde de hemen hemen tüm servisleri sanallaştırmak için kullanılacaktır. Bu kapsamda, gelen trafiği ve konteynırları organize edebilmek için kubernetes konteynır orkestrasyon aracından faydalanılmıştır ("Kubernetes Documentation", t.y.).

2.3. Kod olarak altyapı (Infrastructure as Code – IaC)

Alt yapının manuel olarak yapılandırılması yerine makine tarafından okunabilir tanım dosyaları aracılığı ile sağlandığı ve yönetildiği sistem Kod Olarak Altyapı (Infrastructure as Code – IaC) denmektedir ("Infrastructure as Code", 2023). Önermiş olduğumuz sistemde farklı ihtiyaçları karşılamak üzere oluşturulmuş birçok sistemin bulundurulması ve zaman içerisinde bu sistemlere ekleme yapılabilmesi hedeflenmektedir. Bu bağlamda önerilen sistemde özelleştirilebilir sanal masaüstü görüntüleri oluşturabilmek ve sunabilmek için IaC mimarisinden faydalanılmıştır ("Infrastructure as Code", 2023).

2.4. Dokuman tabanlı veri tabanı

Dokuman tabanlı (NoSQL) veri tabanı, ilişkisel veri tabanından farklı olarak örnekler arasında ilişki kurmak yerine örnekleri bir veri modeli

sayesinde dosya yapısında saklamaktadır. Kullanmış olduğu veri modeli sayesinde hızlı sorgular atabilmekte, yatay ölçeklenebilir şemalar sağlamakta ve diğer veri tabanı tabloları ile ilişkilendirme yapmaya gerek olmadan json ve xml gibi formlarda veri depolayabilmektedir. Çalışmada önerilen sistemde kullanılacak olan uzun vadeli kayıt sistemlerinin saklanması için açık kaynak kodlu ve dokuman tabanlı MongoDB teknolojisinden faydalanılacaktır ("What Is NoSQL?", t.y.).

2.5. Elasticsearch

NoSQL veri tabanları, dosya arama, silme ve düzenleme gibi işlemlerde ilişkisel veri tabanlarına göre daha hızlı olsa da gerçek zamanlı veri işlemleri için yeterli hızlara ulaşamamaktadır ("Elasticsearch vs MongoDB - A Detailed Comparison of Document-Oriented Databases | SigNoz", 2023). Bu kapsamda büyük, dağınık ve tek başlarına anlamsız veri topluluklarında hızlı arama yapabilmek için tam metin indeksleme araçlarından faydalanılmaktadır. Çalışmamızda kısa süreli kayıtlara hızlı erişim ve görüntüleme işlemleri için Apache Lucene temelinde Java programlama dili ile yazılmış Elasticsearch teknolojinden faydalanılacaktır ("Elasticsearch", t.y.).

2.6. Kubevirt

KubeVirt, sanal makinelerin kubernetes üzerinde çalıştırılmasını sağlayan açık kaynaklı bir sanal makine yönetim sistemidir. KubeVirt, bir Kubernetes konteynır orkestrasyon aracı içinde Çekirdek tabanlı Sanal Makine (Kernel-based Virtual Machine - KVM) kullanarak konteynır orkestrasyon aracında yerel sanallaştırma sağlar. KubeVirt ile hem sanallaştırma yönetimini hem de kubernetes konteynır orkestrasyon aracı düzenlemesi yapılabilmektedir. Bu çalışmada kubernetes kümesindeki konteynırların yanında sanal makineleri de tek elden oluşturabilmek ve yönetebilmek amacıyla KubeVirt sisteminden faydalanılmıştır ("Getting to Know Kubevirt", 2018).

3. UYGULAMA

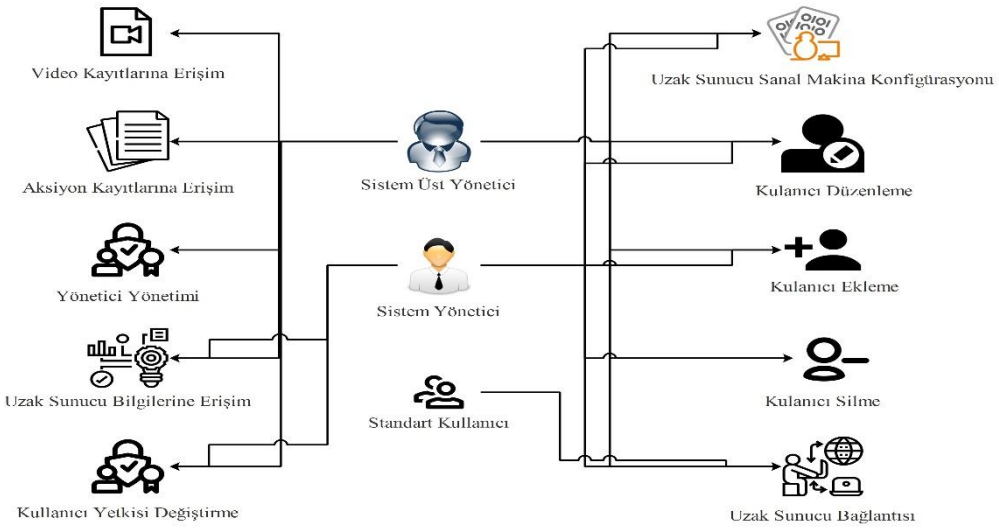
Çalışmamızda, konteynır teknolojileri kullanılarak mikro-servis tabanlı bir PAM sistemi önerilmiştir. Önerilen sistemin güvenliğini artırmak için çok faktörlü doğrulama gerektiren bir giriş sistem geliştirilmiştir. Kod olarak altyapı teknolojisi sayesinde uzak sunucu erişim süreçlerinin, sistem yöneticisi tarafından konfigüre edilmesi sağlanacak bu sayede çalışan eforu azaltılacaktır. Oturum açma ve uzak sunucu bağlantı süreçlerinin tamamı ekran video kayıt sistemi ile tutulacak bu sayede güvenlik ihlalleri rahatlıkla tespit edilebilecektir. Bu sistemin aşamaları Bölüm 3.1, 3.2 ve 3.3'te detaylı olarak anlatılmaktadır.

3.1. Önerilen sistem kullanıcı türleri ve sistem giriş adımları

Çalışma kapsamında önerilen sistemde üç farklı tip kullanıcı bulunmaktadır. Söz konusu kullanıcıların yetkileri dahilinde erişebileceği sistemler Şekil 2 de görülmektedir. Kullanıcıların yetki seviyeleri en az yetki seviyesinden en kapsamlı yetki seviyesine doğru; standart kullanıcı, sistem yöneticisi ve sistem üst yöneticisi olarak sıralanmaktadır. Bir üst yetki seviyesindeki kullanıcı alt seviyelerdeki kullanıcıların yetkilerine de sahip olacaktır. Bu kapsamda standart kullanıcı yalnızca uzak sunucu bağlantısı yapabilecektir. Sistem yöneticisi standart kullanıcı yetkisine ek olarak uzak sunucu bilgilerine erişim, kullanıcı yetkisi değiştirme, kullanıcı ekleme, silme ve düzenleme, video kayıtlarına erişim, uzak sunucu sanal makine konfigürasyonu yetkilerine sahip olacaktır. Sistem üst yöneticisi ise tüm yetkilere ek olarak aksiyon kayıtlarına erişim ve yönetici yönetim yetkilerine sahip olacaktır.

Giriş sisteminin temelde dört ana kontrolü yapması hedeflenmektedir. İlk olarak PAM sisteminde kullanıcının aktif olup olmadığı, daha sonra üçüncü parti uygulamalarla (personel bilgi sistemi, kurumsal kaynak planlama uygulamaları vb.) iletişime geçecek servis sayesinde kullanıcının firmadaki durumu, ardından kullanıcı adı ve şifre doğruluğu, son olarak ise ikinci faktör doğrulması kontrol edilecektir. Sisteme giriş sürecinde gerçekleştirilen bütün kontrol aşamaları Şekil 3 de detaylı olarak görülmektedir.

Kullanıcılar PAM kullanıcı ve şifre bilgileri ile sisteme giriş yapabilmek için talepte bulunacaktır. Giriş talebi sonrasında gerçekleştirilen PAM sistemi aktiflik kontrolü sonucunda kullanıcının pasif olması durumunda kullanıcı tekrardan sisteme giriş yapmaya yönlendirilecektir. Kullanıcının aktif olması durumunda ise üçüncü parti sistem aktiflik kontrolü yapılacaktır. Üçüncü parti sistem aktiflik kontrolü sonucunda aktif olmayan kullanıcılar PAM sistemi tarafından pasif kullanıcı olarak değerlendirilip kayıt defteri bildirimi yapılacaktır. Kontrol sonucunda aktif olarak değerlendirilen kullanıcılara ise şifre kontrolü yapılacaktır. Şifre kontrolü sırasında yanlış şifre girilmesi durumunda kullanıcı tekrar sisteme giriş aşamasına yönlendirilecektir. Şifre kontrolü aşamasında üç defa hatalı giriş yapan kullanıcı sistem tarafından pasif kullanıcı olarak değerlendirilip kayıt defteri bildirimi yapılacaktır. Şifre kontrolünü geçen kullanıcılar için ise ilk giriş ya da son şifre değişikliği kontrolü yapılacaktır.



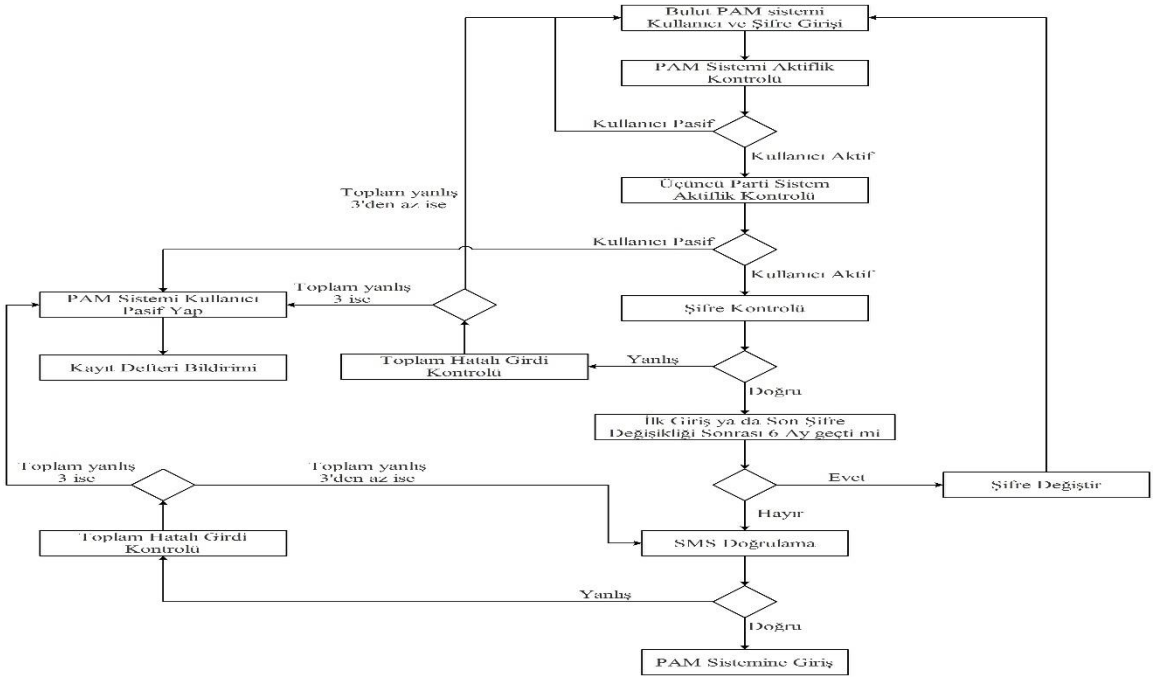
Şekil 2. Kullanıcı durum diyagramları

Kullanıcıların sisteme ilk kez girmesi veya son şifre değişikliği üzerinden altı ay geçmesi durumunda kullanıcıların şifrelerini değiştirerek tekrardan sisteme giriş talebinde bulunması gerekmektedir. Şifre değiştirmesine gerek olmayan kullanıcılar ise SMS doğrulama aşamasına yönlendirilecektir. SMS doğrulaması sırasında hatalı giriş yapan kullanıcıların tekrar doğrulama yapması istenecektir. Üç defa hatalı giriş yapan kullanıcılar sistem tarafından pasif kullanıcı olarak değerlendirilip kayıt defteri bildirimi yapılacaktır. SMS doğrulamasını geçen kullanıcılar ise PAM sistemine giriş yapacaktır.

3.2. Önerilen sistem fonksiyonları

Bu çalışmada PAM sisteminde uzak sunuculara bağlanmak için sanal masaüstüler oluşturulmuş ve bağlantılar bu sanal masaüstüler üzerinden yapılmıştır. Uygulama sistemini özetleyen teknik mimari yapımız Şekil 1 de gösterilmiştir. Tasarlanan sistem üzerinden bağlantı yapmış her bir kullanıcı için farklı sanal masaüstü oluşturulacak ve uzak erişim istenen sunucu için bağlantı gereksinimlerini (vpn, rdp, ERP bağlantısı, ayar dosyası vb.) bu sanal masaüstünde otomatik olarak oluşturulmuştur. Aynı sunucuya yapılmış olan her bir bağlantı için, kullanıcı bilgileri ile ilgili sanal masaüstü çalıştırılmıştır. Bu kapsamda sistem için bir sanal masaüstü yöneticisi olan, açık kaynak kodlu kubernetes, konteynır ve VDI teknolojilerinden faydalanılmıştır.

Tasarlanan PAM sistemi tek bir noktadan yönetilmiştir.



Şekil 3. Önerilen sistem girişi akış diyagramı

Erişim aşamalarında sanal masaüstü kullanılmış olduğu ve bağlantının kişisel bilgisayar yerine sanal masaüstü üzerinden yapılmış olduğu göz önüne alındığında fiziki kaynaklardan ötürü aynı anda yapılabilecek bağlantı sayısının bir limiti olacağı saptanmıştır. Bunun yanı sıra aynı anda yapılabilen bağlantı sayısı, erişim istenen sunucuların gereksinimlerine göre istediği kaynak miktarına bağlı olarak değişiklik göstermiştir. Çalışma kapsamında sistem alt yapısı için dinamik olarak ölçeklenebilir bir mimariye ihtiyaç duyulmuştur. Bu amaçla sistem kaynaklarını ve istek gereksinimlerini yönetebilen açık kaynak kodlu, çalışma başlatmak, çalışma yürütmek gibi işlevleri olan ve bekleyen işlerin sırasını yönetebilen kaynak yönetim yazılımı kubernetes vdi kullanılmıştır.

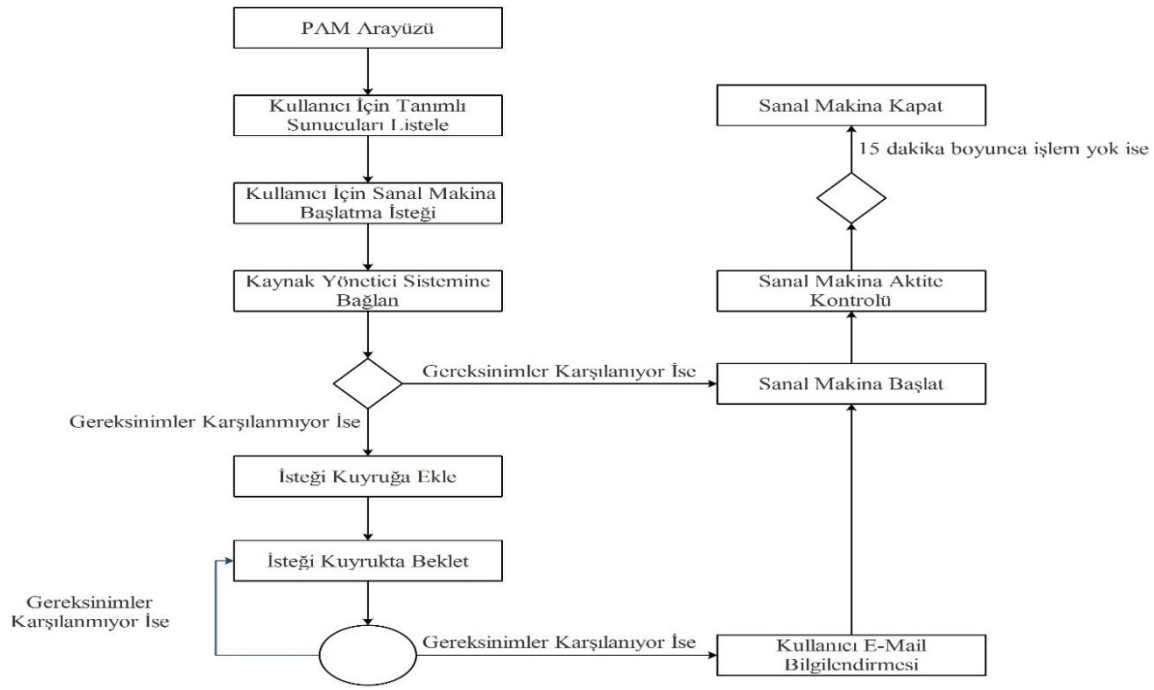
Sistemin güçlü bir kayıt defteri sisteminin olması ve ihlal durumlarının tespit edilebilmesi için Güvenlik Bilgileri ve Olay Yönetimi (Security Information and Event Management - SIEM) teknolojisinden faydalanılmıştır.

Tasarlanan sistemde kısa ve uzun vadeli olmak üzere kayıt işlemlerinin iki farklı şekilde tutulması planlanmıştır. Kısa vadeli kayıt işlemleri son birkaç günün aksiyonlarının tutulduğu sistemi temsil etmektedir. Bu kayıtlarda arama ve görüntüleme işlemlerinin kolay yapılabilmesi için açık kaynak kodlu tam metin indeksleme aracı olan ElasticSearch teknolojisinden faydalanılacaktır. Uzun vadeli kayıt işlemleri ise, aksiyonların daha uzun süreli zaman dilimlerinde tutulduğu sistemi temsil etmektedir. Elasticsearch gibi tam metin indeksleme araçlarının fazla kaynak tüketiminden ötürü, uzun süreli zaman diliminde tutulan aksiyonlar için dokuman tabanlı veri tabanı olan açık kaynak kodlu MongoDB teknolojisinden faydalanılacaktır.

Bu çalışmada olay takip edilebilme kabiliyetini artırmak için, kayıt defterinin yanı sıra ekran videoları da alınmaktadır. Bu kapsamda Java tabanlı "Robot" kütüphanesinden faydalanılmıştır. Bu kütüphanelerden erişimin ve kullanımlarının kolay olması ve video kayıt için isteklerimizi karşılaması nedeniyle yararlanılmıştır. Oluşturulacak video kaydının optimizasyonu için mpeg, h264 vb. sıkıştırma teknolojilerinden faydalanılmıştır.

3.3. Önerilen sistem üzerinden oturum başlatma süreci

Oturum başlatma süreci kullanıcının PAM ara yüzüne giriş yapmasıyla başlar. İlk olarak sistem tarafından kullanıcı için tanımlı sunucular listelenir ve sanal makine başlatma isteğinde bulunulur. Sonrasında otomatik olarak kaynak yönetim sistemine bağlanılır ve sanal makine başlatılması için gereksinimlerin karşılanıp karşılanmadığı kontrol edilir. Gereksinimler karşılanıyor ise sanal makine başlatılır. Aksi durumda sanal makine başlatma isteği kuyruğa alınır ve gereksinimler karşılanana kadar kuyrukta bekletilir. Gereksinimler karşılandığında ise kullanıcıya e-mail yoluyla bilgilendirme yapılarak sanal makine başlatılır. Devam eden süreçte sanal makine aktivite kontrolü yapılarak oturum başlatılır. Oturumun başlatılmasından itibaren kullanıcının on beş dakika herhangi bir işlem yapmaması durumunda sistem tarafından sanal makine kapatılır. Önerilen sistemde kullanıcı oturum başlatma süreci şekil 4 de detaylı olarak gösterilmiştir. Sanal makineler kullanıcı spesifik olarak özelleştirildiğinden IaC alt yapısından faydalanılmıştır. Kullanıcıya özel sanal makineler sunmak için ise terraform konteynır şablonundan faydalanılmıştır.



Şekil 4. Önerilen sistem üzerinden oturum başlatma akış diyagramı

4. SONUÇLAR

Bu çalışmada, yenilikçi yaklaşımlar kullanılarak bir PAM sistemi önerilmiştir. Önerilen bu sistemde platform bağımsız çalışma için konteynır teknolojisinden faydalanılmıştır. Yatay olarak büyümeye olanak sağlamak ve düğümler arası yük dengesi sağlaması amacıyla konteynır orkestrasyon araçları kullanılmıştır. Sistem kullanıcılarının ihtiyaç duyduğu araç setlerinin otomatik olarak sağlanması amacıyla kod olarak alt yapı mimarisi ile geliştirilmiştir. Bu sistemi kullanan kullanıcıların kısa süreli aksiyon kaydı verilerine hızlı ulaşabilmek için tam metin indeksleme aracı, uzun süreli aksiyon ve video kayıtlarının saklanması için doküman tabanlı veri tabanı kullanılmıştır. Sistem aksiyon kayıtlarının raporlanması ve analizi için Güvenlik Bilgileri ve Olay Yönetim sistemi kullanılmıştır. Bu sisteme güvenli giriş yapılabilmesi amacıyla çok faktörlü giriş yönetim sistemi tasarlanarak sisteme entegre edilmiştir. Tüm bu teknolojilerden faydalanılarak uzak sunucu bağlantısı yapacak kullanıcıların, erişim ayarlarının be bağlantı süreçlerinin yönetilmesi için ayrıcalıklı erişim yönetim sistemi geliştirilmiştir.

Tasarlanan sistem, üçüncü parti sistemlerle entegre çalışabilen bir giriş sistemine sahip olması, uzun ve kısa vadeli kayıt sistemleri sayesinde güçlü bir aksiyon analizine olanak tanıması, sanal masaüstü alt yapısından faydalanarak her erişimde izole bağlantı açabilme kapasitesi, anlık ihlal tespitlerine uyumlu olması, erişim süreci boyunca ekran kaydı alabilmesi, sunucu erişim ayarlarının otomatik kurulu olduğu sanal masaüstüler sayesinde kullanım kolaylığı sağlaması ile literatürde tasarlanmış ayrıcalıklı erişim yönetim sistemlerine yeni bir bakış açışı kazandırmaktadır. Tüm bunların tanı sıra konteynır ve kod olarak altyapı teknolojileri sayesinde uzak sunucuya erişimde kullanılabilecek sistemlerin yönetici tarafından oluşturulabilmesi sağlanacak ve oluşturulan bu sistemleri kimlerin kullanabileceği belirlenecektir. Yönetici tarafından oluşturulan bu sistemler kullanılarak, ilgili sunucuya bağlanmak ve o sunucuda işlem yapmak için gerekli olan tüm kurulumlar yönetici tarafından yapılacak, ilgili sunucuya bağlantı için kullanılacak şifreler ise sadece yönetici tarafından bilinecek ve erişimi sağlayacak kişi tarafından bilinmeyecektir. Çok faktörlü doğrulama gerekmesi durumunda ise bu doğrulama önerilen sistem tarafından otomatik olarak yapılacaktır. Örneğin, bir kullanıcının SAP Logon kullanarak bir müşterinin sistemine bağlanması gerektiğini ve bu bağlantı için ise çok faktörlüğü doğrulama barındıran proxy kullanılacağını varsayalım. Önermiş olduğumuz PAM uygulamasında sistem yöneticisi, içerisinde SAP Logon ve ilgili müşterinin giriş bilgilerini barındıran bir sanal makina konfigürasyonu gerçekleştirecektir. Uzak bağlantı yapmak isteyen çalışan, SAP Logon ve proxy kurulumu gibi hiçbir süreçle uğraşmadan sistemi kullanarak bağlantı yapabilecektir. Birçok çalışanın aynı firmaya destek vereceği düşünüldüğünde bu kurulumların normal koşullarda tüm çalışanlar tarafından yapılması gerekmektedir. Bir çalışanın ise farklı firmalara destek verdiği durumlarda, destek verilen firmaların gereksinimlerine bağlı olarak farklı kurulumlar yapması gerekmektedir. Önerilen sistem kullanılarak tüm bu durumlar sistem yöneticisi tarafından tek seferde yapılacaktır. Bu sayede yazılım danışmanlığı yapan firmaların aynı anda birçok kurumun farklı sistemlerine uzaktan destek verme sürecindeki uzak sistemlere bağlanma, bağlantı süreçlerini yönetme, şifre güvenliği ve erişim düzeylerinin belirlenmesi işlemlerinin en az efor ile yapılabilmesini sağlamıştır.

Bu çalışma sonucunda kuyrukta bekletilen kullanıcıların gereksinimleri karşılandığında hangi öncelik sırasına göre iş kuyruğuna ekleneceği noktasında yapay zekâ destekli dinamik bir ölçekleme sistemi ihtiyacı doğmuştur. Bir sonraki çalışmamızda, tasarlanmış olan sistem tarafından toplanan veriler analiz edilecek ve yapay zekâ destekli bir ölçekleme sistemi geliştirilecektir.

BİLGİLENDİRME/TEŞEKKÜR

Bu çalışma, Detay Teknoloji Yazılım Danışmanlık Bilgisayar Hizmetleri Tic. San. A.S Ar-Ge Merkezi bünyesinde yürütülen çalışmaların sonucudur. Desteklerinden dolayı Merkeze teşekkür ederiz.

KAYNAKÇA

Alruwies, M., Mishra, S., Abdul, M., & Alshehri, R. (2021). Identity Governance Framework for Privileged Users. Computer Systems Science and Engineering, 40. https://doi.org/10.32604/csse.2022.019355

Anton, P., & Soetomo, M. (2018). Assessing Privileged Access Management (PAM) using ISO 27001: 2013 Control. 5, 65-76. Annual Conference on Management and Information Technology.

Docker Documentation. (200M.S., 42:25 + +0200). Geliş tarihi 02 Ekim 2023, gönderen Docker Documentation website: https://docs.docker.com/

D'Silva, D., & Ambawade, D. D. (2021). Building A Zero Trust Architecture Using Kubernetes. 2021 6th International Conference for Convergence in Technology (I2CT), 1-8. https://doi.org/10.1109/I2CT51068.2021.9418203

Elasticsearch: The Official Distributed Search & Analytics Engine. (t.y.). Geliş tarihi 02 Ekim 2023, gönderen Elastic website: https://www.elastic.co/elasticsearch

Elasticsearch vs MongoDB - A detailed comparison of Document-Oriented Databases | SigNoz.

(2023, Ocak 20). Geliş tarihi 08 Ekim 2023, gönderen https://signoz.io/blog/elasticsearch-vs-mongodb/

Garbis, J., & Chapman, J. W. (2021). Privileged Access Management. Içinde J. Garbis & J. W. Chapman (Ed.), Zero Trust Security: An Enterprise Guide (ss. 155-161). Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-6702-8_12

Getting to Know Kubevirt. (2018, Mayıs 22). Geliş tarihi 02 Ekim 2023, gönderen Kubernetes website: https://kubernetes.io/blog/2018/05/22/getting-to-know-kubevirt/

Infrastructure as code. (2023). Içinde Wikipedia. Geliş tarihi gönderen https://en.wikipedia.org/w/index.php?title=Infrastructure_as_code&oldid=1176394945

Ionita, V. (2023). Privileged access management for databases. Geliş tarihi gönderen https://aaltodoc.aalto.fi:443/handle/123456789/122872

Kubernetes Documentation. (t.y.). Geliş tarihi 02 Ekim 2023, gönderen Kubernetes website: https://kubernetes.io/docs/home/

Lang, D., Jiang, H., Ding, W., & Bai, Y. (2019). Research on Docker Role Access Control Mechanism Based on DRBAC. Journal of Physics: Conference Series, 1168(3), 032127. https://doi.org/10.1088/1742-6596/1168/3/032127

Preuveneers, D., & Joosen, W. (t.y.). Federated Privileged Identity Management for Break-the-Glass: A Case Study with OpenAM.

Sindiren, E., & Ciylan, B. (2018). Privileged Account Management Approach for Preventing Insider Attacks.

Sindiren, E., & Ciylan, B. (2019). Application model for privileged account access control system in enterprise networks. Computers & Security, 83, 52-67. https://doi.org/10.1016/j.cose.2019.01.008

Steinhoff, M. (2020). Using Software Containers for Privileged Access Management in Cloud Environments: A Novel Approach to Handle Access Management for Cloud-based Networks. Nordic and Baltic Journal of Information & Communications Technologies, 297-310. https://doi.org/10.13052/nbjict1902-097X.2020.013

Tabrizchi, H., & Kuchaki Rafsanjani, M. (2020). A survey on security challenges in cloud computing: Issues, threats, and solutions. The Journal of Supercomputing, 76(12), 9493-9532. https://doi.org/10.1007/s11227-020-03213-1

Tep, K. S., Martini, B., Hunt, R., & Choo, K.-K. R. (2015). A Taxonomy of Cloud Attack Consequences and Mitigation Strategies: The Role of Access Control and Privileged Access Management. 2015 IEEE Trustcom/BigDataSE/ISPA, 1, 1073-1080. https://doi.org/10.1109/Trustcom.2015.485

Tran, L. (2020). Privileged Access Management for System to System communications. Geliş tarihi gönderen https://aaltodoc.aalto.fi:443/handle/123456789/46232

What Is NoSQL? NoSQL Databases Explained. (t.y.). Geliş tarihi 02 Ekim 2023, gönderen MongoDB website: https://www.mongodb.com/nosql-explained

Ylonen, T., Turner, P., Scarfone, K., & Souppaya, M. (2015). Security of Interactive and Automated Access Management Using Secure Shell (SSH) (Sy NIST IR 7966; s. NIST IR 7966). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.IR.7966

BİLGİSAYAR BİLİMLERİ VE TEKNOLOJİLERİ DERGİSİ
JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGIES
http://dergipark.org.tr/tr/pub/bibted

Araştırma Makalesi

Gerçek Zamanlı Gömülü Sistemlerde Enerji Tüketiminin Azaltılması İçin Teknikler

Abdullah ELEWİ*[1], Ayşegül YAMAN[1], Sibel KAPLAN[1], Ahmed Abd ALKADER[2]

[1]Mersin Üniversitesi, Bilgisayar Mühendisliği Bölümü, Mersin, Türkiye
[2]Hatay Mustafa Kemal Üniversitesi, Hassa Meslek Yüksekokulu, Hatay, Türkiye

Anahtar Kelimeler:
Gerçek zamanlı sistem
Enerji tüketimi
DVFS
DPM
MCRTsim

ÖZ

Gerçek zamanlı gömülü sistemler, basit gadget'lardan karmaşık aviyonik sistemlere kadar modern hayatımızın her detayına dahil edilmiştir. Güç, gerçek zamanlı gömülü sistemlerin tasarımı ve işletiminde optimizasyon için önemli ölçütlerden biridir. İşlemci biriminde enerji tüketimini azaltmak için başlıca kullanılan teknikler; işlemcinin yavaşlama faktörlerine dayalı dinamik voltaj/frekans ölçeklendirmesi (DVFS) ve dinamik güç yönetimi (DPM)dir. Bu çalışmada kaynaklara erişim protokolleri ve gerçek zamanlı zamanlama algoritmaları kullanarak; DVFS içerisinde yer alan MaxSpeed (MS) ve kritik bölüm maksimum hız (CSMS) teknikleri, ve DPM incelenmiş ve yapılan örnekler MCRTsim programı üzerinde çalıştırılıp sonuçları alınıp karşılaştırılmıştır. Elde edilen sonuçlar, bu tekniklerin uygulanmasının, kullanılan örneklerde enerji tüketimini %27'ye kadar azaltabileceğini göstermiştir.

Energy Consumption Reduction Techniques in Real-Time Embedded Systems

Keywords:
Real-time system
Energy consumption
DVFS
DPM
MCRTsim

ABSTRACT

Real-time embedded systems are incorporated in every detail of our contemporary life from simple gadgets to avionics complex systems. Power is one of the important criteria for optimization in the design and operation of real-time embedded systems. Mainly used techniques to reduce energy consumption in the processor unit are dynamic voltage/ frequency scaling (DVFS), based on processor slowdown capabilities, and dynamic power management (DPM). In this study, using resource access protocols and real-time scheduling algorithms; MaxSpeed (MS) and critical section maximum speed (CSMS) techniques of DVFS and DPM were examined with examples run on the MCRTsim simulation program to obtain and compare results. The obtained results showed that applying such techniques can reduce energy consumption to ratios of 27% for the utilized examples.

* **Sorumlu Yazar**

*(elewi@mersin.edu.tr) ORCID ID 0000 – 0001 – 9774 – 5292
(aysegulyaman@mersin.edu.tr) ORCID ID 0000 – 0002 – 6972 – 6657
(sibelkaplan@mersin.edu.tr) ORCID ID 0000 – 0003 – 3299 – 4882
(akader@mku.edu.tr) ORCID ID 0000-0002-0538-7924

e-ISSN: 2717-8579
Geliş Tarihi: 22/09/2023; Kabul Tarihi: 27/12/2023

1. GİRİŞ

Gömülü bir sistem, bağımsız bir sistem olarak veya büyük bir sistemin parçası olarak özel bir işlevi yerine getirmek üzere tasarlanmış yazılıma sahip mikroişlemci tabanlı bir bilgisayar donanım sistemidir. Çekirdekte, gerçek zamanlı işlemler için hesaplama yapmak üzere tasarlanmış entegre bir devre bulunur. Karmaşıklıklar, tek bir mikro denetleyiciden bağlı çevre birimleri ve ağları olan bir işlemci paketine kadar uzanır. Gömülü bir sistemin karmaşıklığı, tasarlandığı göreve bağlı olarak önemli ölçüde değişir (Lipskoch ve Ark., 2007).

Gerçek zamanlı işletim sistemleri genellikle, gömülü sistemlerde bulunur ve çalışmasında zaman kavramının çok önemli olduğu yerlerde kullanılır. Elektronik bir cihazın kontrol kartında bulunan ve sistem içerisinde görev alan yapıların belli bir işlem sırası vardır. Gerçek zamanlı işletim sistemleri, gerçek zaman ve işletim sistemi olarak iki parçadan oluşur. Bu sistemler, çoklu görev bilincinde çalışırlar. İşlemler, yapılarında bulunan bir çekirdek üzerinden gerçekleştirilir. Normal işletim sistemlerinden farkı, işlem önceliği yerine zaman önceliğine sahip olmalarıdır. Kendisine verilen görevleri, birbirleri arasında çok hızlı bir şekilde geçiş sağlayarak yerine getirerek, bütün işlemler aynı anda gerçekleşiyor izlenimi verirler. Gerçek zamanlı bir sistem, iyi tanımlanmış, sabit zaman kısıtlamaları olan zamana bağlı bir sistemdir (Liu, 2000; Laplante ve Ovaska, 2012).

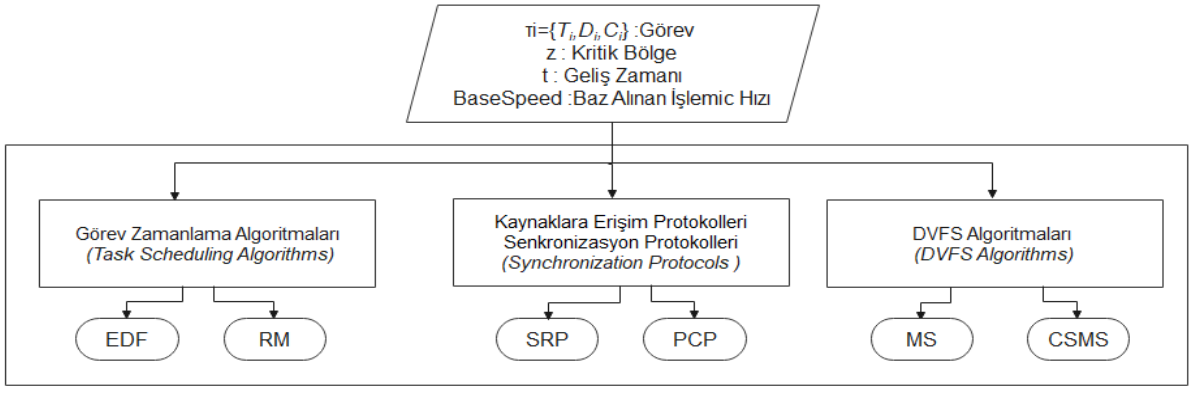
Gerçek zamanlı gömülü sistemlerde güç tüketiminde enerji tasarrufu dikkate alınması gereken önemli bir faktördür. Bu sistemlerde güç tüketimini azaltmanın iki ana yolu vardır: işlemcinin yavaşlama faktörlerine dayalı dinamik voltaj ve frekans ölçeklendirmesi (DVFS) ve dinamik güç yönetimi (DPM)dir (Schmitz ve Ark., 2005). Frekans veya voltaj ölçeklendirme kullanarak yavaşlatma, güç tüketimini azaltmada daha etkilidir. Bir işlemcinin frekansını ve voltajını ölçeklendirmek, bir işin yürütme süresinde bir artışa yol açar. Enerjiyi en aza indirme hedefimize ulaşmak için zamanı ve gücü mantıklı bir şekilde yönetmek zorundayız.

DVFS, besleme voltajını ve çalışma frekansını düşürerek enerji kaybını azaltmak için işlemcilerin donanım özelliklerinden yararlanmada önemli bir teknik olmuştur. DVFS algoritmalarının, genel amaçlı sistemlerde en yüksek işlemci gücünde çalışırken, önemli ölçüde enerji tasarrufu sağlayabildiği gösterilmiştir (Pillai ve Shin, 2001; Awadalla ve Elewi, 2016; Saad ve Ark., 2013).

Bu yazıda, sabit yavaşlama faktörlerinin hesaplanması yoluyla sistem düzeyinde güç yönetimine odaklanılmıştır. Literatürde birden fazla DVFS algoritması mevcuttur. Bu çalışmada; DVFS algoritmalarından MS (MaxSpeed) ve CSMS (Critical Section Maximum Speed) algoritmaları incelenmiştir. Bu algoritmalar incelenirken, zamanlama algoritması olarak Monoton Oran (Rate Monotonic-RM) ve Erken Biten Önce (Earliest Deadline First-EDF) kullanılmıştır (Palamut ve Ark., 2019; Yıldırım ve Ark., 2020). Paylaşılan kaynaklara erişmek için Yığın Kaynak Politikası öncelikli tavan protokolü(Priority Ceiling Protocol - PCP) (Cheng ve Ras, 2007) ve (Stack-Resource Policy - SRP) (Baker, 1991) algoritmaları kullanılmıştır. Sistem tasarımı için yapılan örnekler MCRTsim (Wu ve Huang, 2017) programı üzerinde çalıştırılıp sonuçları alınmıştır. MCRTsim programı java tabanlı ve açık kaynaklı bir uygulamadır.

2. SİSTEM İÇİN GEREKEN BİLEŞENLER

Bu bölümde, tek işlemcili gerçek zamanlı sistem için gerekli girdi parametrelerini ve uygulama için kullanılacak algoritmaları tanıtıyoruz ve formüle ediyoruz. Şekil 1 bu bileşenleri özetlemektedir.



Şekil 1: Tek İşlemcili Gerçek Zamanlı Sistem İçin Gereken Bileşenler

2.1. Sistem Girdileri

Sistem modeli için öncelikli olarak gerçek zamanlı işletim sisteminde kullanılan bazı terimlerin karşılığına bakmalıyız.

- τ Görev *(Task)*: Sistem işlevlerini birlikte sağlayabilen bir dizi
- İş *(Job)*: İşlemciye atanan küçük iş parçası
- *r* Görevin geliş zamanı (arrival time)
- *T (Periyod)*: Görevin periyodu

· *D* İşin Zaman Sınırı *(Relative deadline)*: İşin tamamlanması gereken zaman
· *C* En Kötü Çalışma Süresi *(Worst case excuation time: WCET)*: İşin bitmesi için geçen süre
. *z* Kritik Bölüm *(Critical Section)*: Eşzamanlı çalışan ve aynı kaynağa ihtiyaç duyan iki görevden birinin kaynağı işi bitene kadar erişimi kilitlediği zaman aralığıdır.
. *n* Görev Sayısı
. *U* İşlemci Kullanım Yüzdesi
. *S* Semafor

Bir görev seti $\Gamma=\{\tau_1, \tau_2, \tau_3 \ldots, \tau_n\}$ şeklinde görevlerden oluşur ve her bir görev ise $\tau_i=\{T_i,D_i,C_i\}$ şeklinde parametrelere sahiptir. T_i görevin periyodu, D_i görevin zaman sınırı, C_i ise çalışma süresi (Worst case excution time: WCET) parametrelerini ifade etmektedir. *($D_i<=T_i$ olmalıdır)*. Görevin her çağrılışına iş *(Job)* denir. Bir görev setinin gerçekleştirilebilmesi için görevlerin her birinin kendi zaman sınırısında tamamlanması gereklidir. Bunun için $U = \sum_{i=1}^{n} \frac{C_i}{T_i} \leq 1$ formülü sağlanmalıdır. Sistem, görevler tarafından karşılıklı ayrıcalıklı bir şekilde erişilen bir dizi paylaşılan kaynağa sahiptir. Paylaşılan bir kaynağa erişim izni verilen görev, kritik bölümde yürütülür. τ_i görevinin k'inci kritik bölümü $z_{i,k}$ olarak temsil edilir. Paylaşılan bir kaynağı kullanması gereken bir görev, o kaynak daha düşük öncelikli bir görev tarafından kullanılıyorsa kaynak serbest bırakılana kadar engellenir. Kaynağı tutan göreve Engelleyen Görev *(Blocking Task)* denir (Jejurikar ve Gupta, 2002; Jejurikar ve Gupta, 2006).

Bir görevin engellendiği süre Görev Engelleme Süresi *(Task Blocking Time)* olarak adlandırılır. Belirtilen görev bilgileri ve belirli bir kaynak erişim protokolü ile bir görev için maksimum engelleme süresi hesaplanabilir. B_i, belirli bir kaynak erişim protokolü altında τ_i görevi için maksimum engelleme süresidir (Jejurikar ve Gupta, 2002).

2.2. Görev Zamanlama Algoritmaları (Task Scheduling Algorithms)

Gerçek zamanlı gömülü bir sistemde DPM/DVFS'nin enerji tüketimi azaltmak, faydalarını gerçekleştirmek için işletim sisteminin görev zamanlama algoritması ile sıkı bir şekilde bağlantılı olması gerekir. 1973 yılında Liu ve Layland (1973), optimal dinamik öncelik zamanlama algoritması olarak Erken Biten Önce (Earliest Deadline First-EDF) algoritması ve optimal sabit öncelik zamanlama algoritması olarak Monoton Oran (Rate Monotonic-RM) algoritmasını sunmuştur. RM algoritmasında görev önceliği görev süresinin uzunluğuna göre verilir. Kısa süreli görevler daha yüksek çalışma önceliğine sahiptir. EDF, görevleri son teslim tarihlerine göre sıralayan dinamik bir öncelik zamanlayıcıdır.

Gerçek zamanlı sistemde bir görev setinin uygun bir şekilde zamanlaması için aşağıdaki koşulları sağlaması gerekir (Liu ve Layland, 1973).

$$\text{EDF için } U = \sum_{i=1}^{n} \frac{C_i}{T_i} \leq 1 \quad (1)$$

$$\text{RM için } U = \sum_{i=1}^{n} \frac{C_i}{T_i} \leq n(\sqrt[n]{2} - 1) \quad (2)$$

2.3. Kaynak Erişim Protokolleri (Resource Access Protocols)

Kaynak erişim protokolleri birden fazla görevin doğru çalışabilmesi için aynı kaynak veya kaynaklara ihtiyaç duyması halinde yaşanan sorunları çözmek için geliştirilmiş tekniklerdir. Protokolün temeli olan "Öncelik Değiştirme Yaklaşımı"na dayanmaktadır (Palamut ve Ark., 2019). Çalışmamızda Öncelikli Tavan Protokolü (Priority Ceiling Protocol-PCP) ve Yığın-Kaynak Politikası (Stack-Resource Policy -SRP) protokolleri kullanılmıştır.

Öncelikli Tavan Protokolü (PCP), Öncelik Kalıtım Protokolü (Priority Inheritance Protocol-PIP)' ye benzer ve aynı zamanda önleyici zamanlamaya dayalıdır. PCP ayrıca aşağıda istenen özellikler: (1) kilitlenmeleri önler; ve (2), zincirleme engellemeyi önler, böylece yüksek öncelikli bir görev, kritik bölümde kendini askıya alsa bile, en fazla bir düşük öncelikli görev tarafından engellenebilir. PCP, PIP'nin neden olduğu doğrudan engelleme ve doğrudan engellemeye ek olarak üçüncü bir engelleme türü olan tavan engellemeyi sunar. Kilitlenmelerin (deadlocks) ve zincirleme blokajların önlenmesi için tavan blokajı gereklidir (Cheng ve Ras, 2007; Buttazzo, 2011).

PCP'nin temel önermesi aşağıdaki kurallara dayanmaktadır: (1) daha yüksek öncelikli bir τ görevini engelleyen daha düşük öncelikli bir görev, τ görevinin önceliğini devralır ve yalnızca görev, sahip olduğu semaforları serbest bıraktığında önceliği normal değerine geri yüklenir. (2) Bir τ görevi yalnızca bir S semaforunu kilitleyebilir: (a) S semaforu henüz kilitlenmemişse ve (b) τ görevinin önceliği tüm semaforların öncelik tavanlarından daha büyükse. Semaforlar şu anda τ görevi dışındaki görevler tarafından kilitlenmiştir. Bir semaforun öncelik tavanı, o semaforu herhangi bir zamanda kilitlemek isteyenlerin en yüksek öncelikli görevi olarak tanımlanır (Sha ve Goodenough, 1990).

Yığın-Kaynak Politikası (Stack-Resource Policy - SRP) Baker (1991) tarafından ortaya sunulan bir tekniktir. Paylaşılan kaynaklar erişmek için sunulmuştur. Öncelikli tavan protokolüne (PCP) ek olarak; çok birimli kaynakların kullanımına izin verir, Dinamik öncelik zamanlamasını destekler ve çalışma zamanı yığın tabanlı kaynakların paylaşımına izin verir (Buttazzo, 2011).

2.4. Enerji Tüketiminin Azaltılması Teknikleri

İşlemcilerin hızı arttıkça enerji tüketimi de arttı. Bir işlemcinin enerji tüketimini azaltmak önemli bir araştırma konusu haline geldi. Bu amaçla DPM ve DVFS teknikleri geliştirilmiştir.

İşlemci için güç tüketimi denklem (3)'te modellenmiştir (Wu ve Huang, 2017).

$$P(f) = \alpha f^{\gamma} + \beta \quad (3)$$

Burada P güç, f frekans (hız), α, β, ve γ işlemci ile ilgili sabitlerdir. Örneğin, Marvell Xscale PCA 270 işlemci için α =1.52, β =0.08 ve γ =3'tür (Wu ve Huang, 2017). Daha sonra, T zaman periyodundaki enerji tüketimi, güç tüketiminin T ile çarpımı olarak basitçe modellenebilir.

DPM, işlemci boştayken basitçe kapatmak anlamına gelir. DVFS algoritmalarında işlemcinin her bir çekirdeğinin var olan en düşük ve en yüksek hızları arasında çalıştığı kabul edilir.

Birçok DVFS algoritması mevcuttur. DVFS algoritmalarından biri olan MaxSpeed (MS) algoritmasında işlemci tüm görevler için en yüksek hızda çalıştırır. Kritik Bölümde En Yüksek Hız (Critical Section Maximum Speed -CSMS) Algoritmasında görevlerin kritik olan ve kritik olmayan bölgelerinin farklı hızlarda çalıştırılması amaçlanmıştır. Bütün görevlerin kritik bölümleri işlemcinin en yüksek hızında çalıştırılmakta iken kritik olmayan bölgeler için yavaşlama faktörleri hesaplanmaktadır. Yani bu algoritmada iki farklı hız kullanılır (Maximum Speed ve baseSpeed). Burada önemli olan tüm görevlerin son teslim süresinden önce sonlandırılmasıdır (Jejurikar ve Gupta, 2002).

3. ENERJİ TÜKETİMİNİN AZALTILMASI TEKNİKLERİNİN MCRTsim SIMÜLASYONU ÜZERİNDE UYGULAMA ANALIZI

DVFS bir enerji tasarrufu tekniğidir, bu özelliği besleme voltajını ve çalışma frekansını düşürerek çekirdeğin enerji kaybını azaltarak elde eder.

MCRTsim, tek işlemcili, çok işlemcili ve çok çekirdekli işlemcili gerçek zamanlı sistemler için açık kaynaklı bir görev zamanlama simülatörü olarak sunulmuştur (Wu ve Huang, 2017). MCRTsim kullanarak mevcut zamanlama algoritmalarının yanı sıra senkronizasyon protokollerinin performansını kolayca değerlendirebiliriz. Ayrıca, MCRTsim, minimum çabayla yeni zamanlama ve senkronizasyon protokollerinin tasarımını desteklemek için bir Java sınıf kitaplığı da içerir. MCRTsim'in bir diğer önemli özelliği, DVFS etkin işlemcilerin desteklenmesidir, böylece enerji duyarlı zamanlama algoritmalarının ve senkronizasyon protokollerinin yeteneklerinin daha iyi anlaşılması sağlanır.

Bu bölümde MCRTSim simülatör üzerinde, zamanlama algoritmaları olarak RM/EDF ve kaynak erişim protokolleri olarak da PCP/SRP algoritmaları ile birlikte DVFS/DPM tekniklerinin kullanımları örneklendirilmiş ve algoritmalar enerji verimliliği açısından analiz edilmiştir. Örneklendirmeler için bir paylaşılan kaynak ve iki periyodik görevden oluşan görev seti kullanılmıştır (Jejurikar ve Gupta, 2002). Aşağıdaki çalıştırılan tüm uygulamalarda bu tasarım kullanılmıştır.

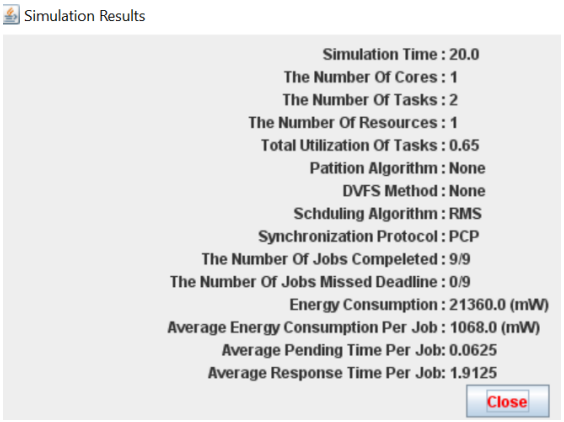
$\tau_i = \{r_i, T_i, D_i, C_i\}$

$\tau_1 = \{0.5, 4, 4,1\}$ $\tau_2 = \{0, 5, 5, 2\}$

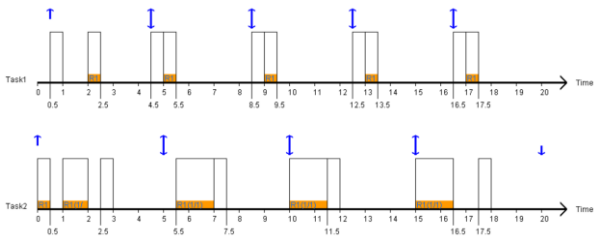
τ_1 görevi için kritik bölüm [0.5, 1] ve τ_2 için [0, 1.5] değerindedir. Kritik bölümler görevleri bloke edebilir ve görevler için maksimum engelleme süresi max hızda B_1 = 1.5 ve B_2 = 0'dır. İşlemcinin maksimum tam hızı (%100) 1000 birim, ve {0, 100, 200, 300, 400, 500, 600, 700 ,800, 900, 1000} birim hızlar çalışabilir. Bu örnek için EDF kullanarak ve hiç ortak kaynak kullanmadan, yavaşlama (slowdown) faktörü U = (2/5) + (1/4) = 0.65 şeklinde hesaplanır.

3.1. DPM/DVFS metodunu RM ve PCP ile Uygulama

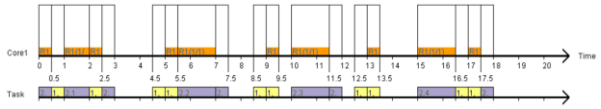
MCRTsim üzerinde verilen konfigürasyon için zamanlama algoritması EDF, DVFS metodu kullanılmadan, kaynak erişim protokolü SRP olmak üzere elde edilen simülasyon sonuçları ve zamanlama grafikları Şekil 2-a, Şekil 2-b ve Şekil 2-c de verilmiştir. Şekil 2-b, her görevi tek başına gösterirken, şekil 2-c bunları Gantt şeması olarak birlikte göstermektedir.



Şekil 2-a: DVFS Methodu Kullanılmadan RM-PCP MCRTSim Simülasyon Sonuçları



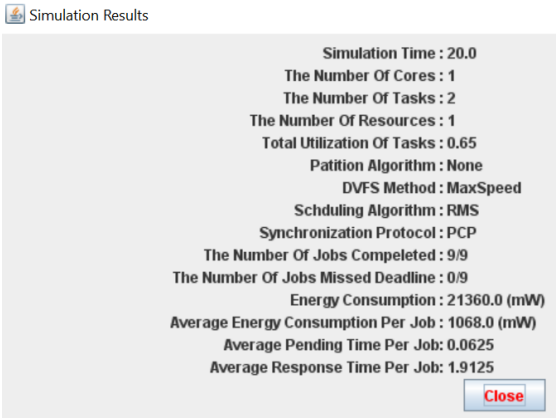
Şekil 2-b: Her görev için MCRTSim Zamanlama Simülasyonu



Şekil 2-c: MCRTSim Zamanlama Simülasyonu

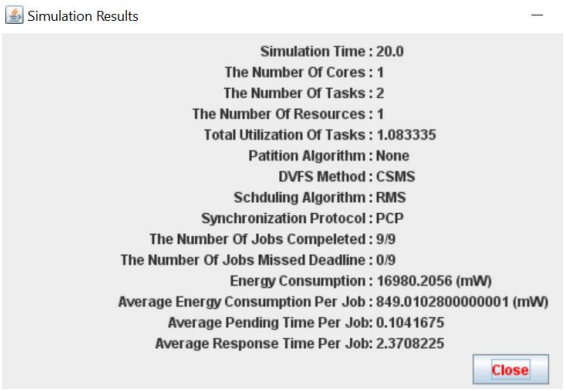
Şekiller kontrol edildiğinde, τ_1'in τ_2 görevini her geldiğinde önleyen sabit bir yüksek önceliğe sahip olduğu ve kritik bölüme girmesi gerektiğinde bloke olabileceği görülebilir.

Bu örnekte, herhangi bir güç azaltma tekniği kullanılmadan enerji tüketimi 21360 mW'dir. Buradaki işlemci, maksimum hız olan 1000 (%100) olan tek temel hızı üzerinde çalışmaktadır.
Ayrıca, işlemci Denklem 3'te β olarak gösterilen statik güç nedeniyle işlemci boştayken bile enerji tüketir. İşlemciyi boştayken kapatan DPM dahil edilirse, enerji tüketimi 21230- (7*80) = 20800 mW'a düşürülebilir.
Her zaman maksimum 1000 (%100) işlemci hızında çalışan MCRTsim'de sunulan MaxSpeed algoritması kullanıldığında tamamen aynı sonuçlar ve zamanlamalar elde edilecektir. Şekil 3-a, MaxSpeed kullanıldığında simülasyon sonuçlarını göstermektedir. Şekil 3-b ve 3-c, sırasıyla şekil 2-b ve 2-c ile tamamen aynıdır.

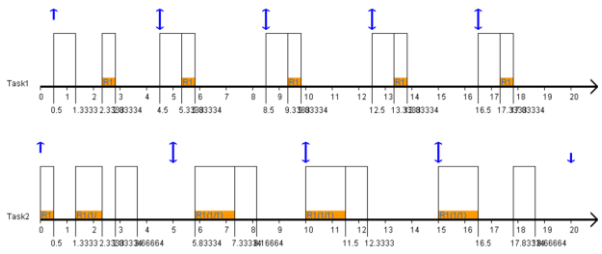


Şekil 3-a: MaxSpeed-RM-PCP MCRTSim Simülasyon Sonuçları

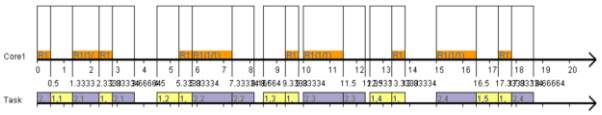
CSMS algoritması kullanılan bu konfigürasyonda kritik olmayan bölgelerdeki görevler 600 (%60) birim hızla çalıştırılırken, kritik bölümlerdeki görevler maksimum hız olan 1000 birim (%100) hızla çalıştırılmıştır. Kritik bölümlerde hız artırıldığı için enerji tüketimi de aynı oranda artmıştır ve 16980.2 (mW) olarak gözlemlenmiştir. Toplam çalışma süreside hızla ters orantılı olarak değişiklik göstermiştir.
Şekil 4-a, 4-b ve 4-c simülasyon sonuçlarını ve zamanlamaları göstermektedir. DPM de etkinleştirilirse, enerji tüketimi 16980.2- (4*80) = 16660.2 mW'a düşürülebilir.



Şekil 4-a: CSMS-RM-PCP MCRTSim Simülasyon Sonuçları



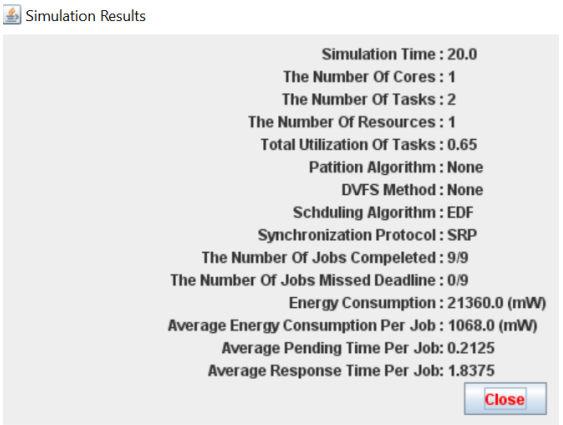
Şekil 4-b: Her görev için MCRTSim Zamanlama Simülasyonu



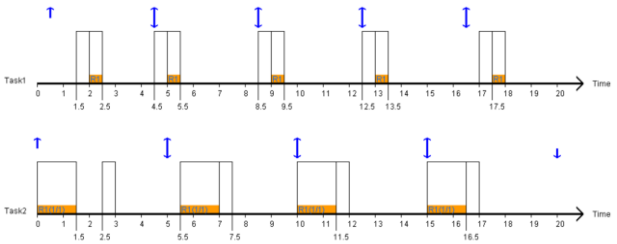
Şekil 4-c: MCRTSim Zamanlama Simülasyonu

3.2. DPM/DVFS metodunu EDF ve SRP ile Uygulama

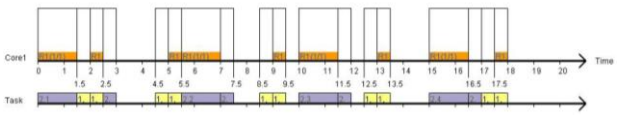
EDF'nin her işin son teslim tarihi ile ilgili dinamik bir önceliği vardır. Ayrıca, görev örneği (iş) geldiğinde SRP'de engelleme olmaktadır. Bu durum, şekil 5-b ve 5-c'de açıkça görülebilir.
EDF-SRP algoritması, DVFS desteklenmediğinde veya MaxSpeed tekniği kullanıldığında RM-PCP ile aynı enerjiyi tüketir.



Şekil 5-a: DVFS Methodu Kullanılmadan EDF-SRP MCRTSim Simülasyon Sonuçları



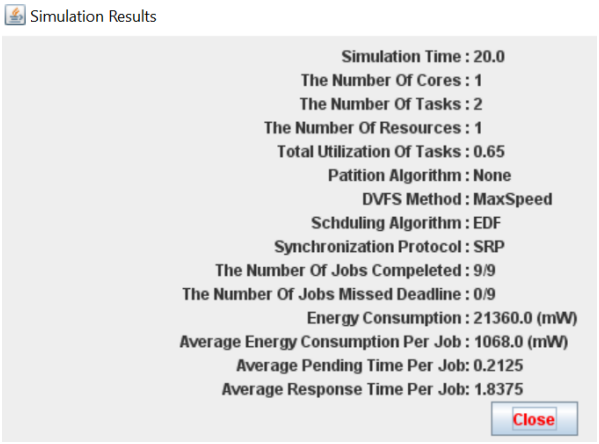
Şekil 5-b: Her görev için EDF-SRP MCRTSim Zamanlama Simülasyonu



Şekil 5-c: EDF-SRP MCRTSim Zamanlama Simülasyonu

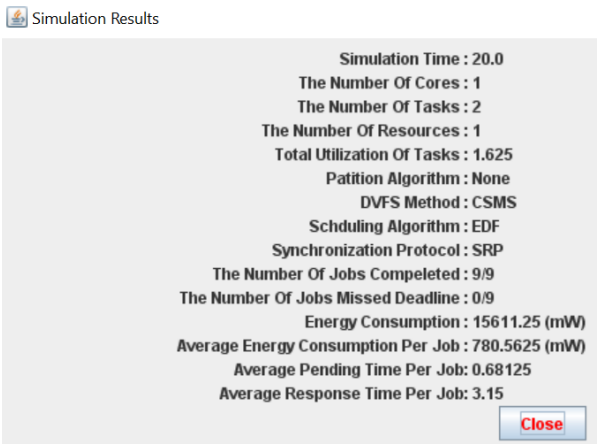
Oluşturulan konfigürasyon DVFS metodu kullanılmadan EDF zamanlayıcısı ve SRP kaynak erişim protokolü ile 1000 birim baz hızında çalıştırıldığında PCP için t=0 anında önceliği düşük olan τ_2 görevi gelir, SRP'den dolayı t=0.5 anında önceliği yüksek olan τ_1 görevi geldiği zaman işlem sırası τ_1 ile devam eder. DVFS metodunun kullanılmadığı bu durumda enerji tüketimi 21360 (mW) olarak gözlemlenmiştir. DPM etkinleştirilirse, enerji tüketimi de 20800 mW'a düşürülebilir.

MCRTsim üzerinde verilen konfigürasyon için zamanlama algoritması EDF, DVFS metodu olarak MaxSpeed, kaynak erişim protokolü SRP olmak üzere elde edilen sonuçlar ve grafik Şekil 6-a verilmiştir. Zamanlamalar şekil 5-b ve 5-c ile tamamen aynıdır.

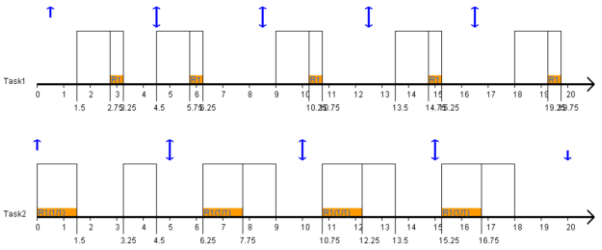


Şekil 6-a: MaxSpeed-EDF-SRP MCRTSim Simülasyon Sonuçları

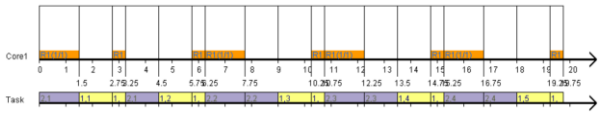
MCRTsim üzerinde verilen konfigürasyon için zamanlama algoritması EDF, DVFS metodu olarak CSMS, kaynak erişim protokolü SRP olmak üzere elde edilen sonuçlar ve grafik Şekil 7-a, Şekil 7-b ve Şekil 7-c de verilmiştir.



Şekil 7-a: CSMS-EDF-PCP MCRTSim Simülasyon Sonuçları



Şekil 7-b: Her görev için EDF-SRP MCRTSim Zamanlama Simülasyonu



Şekil 7-c: EDF-SRP MCRTSim Zamanlama Simülasyonu

CSMS-EDF-SRP için enerji tüketimi 15611,25 mW'dir. DPM de kullanılırsa, enerji tüketimi 15611,25 - (0,25* 80) = 15591,25 mW'a düşürülebilir. Burada kritik bölümler maksimum 1000 (%100) hızda, kritik olmayan bölümler ise 400 (%40) hızda yürütülür.

Simülasyon sonuçları ve zamanlama çizelgeleri karşılaştırıldığında kaynak erişim protokolünü değiştirince sonuçların değişmediği sadece görevlerin kritik bölümde çalışma sırasının değiştiği görülmektedir.

Aynı konfigürasyon üzerinde farklı DVFS Metotları çalıştırıldığında elde edilen sonuçlar Tablo 1'de gösterilmiştir.

Tablo1 de seçilen DVFS algoritmalarına göre elde edilen enerji tüketimi, çalışma süreleri ve çalıştırılan işlemci hızları gösterilmektedir. Hız artınca çalışma sürelerinin azaldığı fakat enerji tüketiminin arttığı görülmektedir. Karşılaştırılan algoritmalar göz önünde bulundurulduğunda CSMS algoritmasının zaman- hız tüketimi açısından daha optimal bir sonuç verdiği görülmüştür.

Verilen örnekte öncelik değişmediği için zamanlama algoritması olarak RM kullanıldığında da aynı sonuçlar elde edilmiştir.

Tablo 1: DVFS algoritmalarına ve verilen işlemci hızına göre elde edilen çalışma süreleri ve enerji tüketimi

DVFS Metodu	Zamanlama Algoritması	Çalışma Süresi		İşlemci Hızı		Enerji Tüketimi (mW)		Enerji Tüketimi Azaltılması (%)	
		Kritik Bölüm	Kritik Olmayan Bölüm	Kritik Bölüm	Kritik Olmayan Bölüm	DPM Olmadan	DPM	DPM Olmadan	DPM
Yok/MaxSpeed	RM	τ_1:0.50 τ_2:1.5	τ_1:0.50 τ_2:0.5	%100	%100	21360	20800	%0	%2.6
Yok/MaxSpeed	EDF	τ_1:0.5 τ_2:1.50	τ_1:0.5 τ_2:0.5	%100	%100	21360	20800	%0	%2.6
CSMS	RM	τ_1:0.5 τ_2:1.50	τ_1:0.833 τ_2:0.833	%100	%60	16980.2	16660.2	%20.5	%22
CSMS	EDF	τ_1:0.5 τ_2:1.50	τ_1:1.25 τ_2:1.25	%100	%40	15611.25	15591.25	%26.9	%27

4. SONUÇ

Bu çalışmada tek işlemcili gerçek zamanlı sistemler üzerinde DVFS algoritmaları ele alınmış ve Java tabanlı MCRTsim uygulaması kullanılarak sonuçlar elde edilmiş ve sonuçlar karşılaştırılıp analiz edilmiştir.

Dinamik voltaj ve frekans ölçekleme (DVFS) tekniğinde amaç işlemcinin voltajını ve frekansını dinamik olarak ayarlayarak dinamik güç tüketimini azaltmayı amaçlamaktır. Elde edilen sonuçlar incelendiğinde de bu amacın gerçekleştirildiği ve aynı konfigürasyonun için enerji tüketiminin %27'ye kadar azaldığı gözlemlenmiştir.

Kaynak Erişim Protokolü olan PCP ve SRP arasında enerji tüketimi açısından bir farklılık görülmemiştir sadece kritik bölümlerde görevlerin çalışma sırası değişmiştir. Aynı Şekilde zamanlama algoritmaları olan EDF ve RM arasında da enerji tüketimi açısından bir farklılık görülmemiştir.

KAYNAKLAR

Awadalla, M. & Elewi, A. (2016). Enhanced PSO approach for real time systems scheduling. International Journal of Computer Theory and Engineering 8(4), 285-289. https://doi.org/10.7763/ijcte.2016.v8.1059

Baker, T. P. (1991). Stack-based scheduling of realtime processes. Real-Time Systems 3(1), 67-99.

Buttazzo, G. C. (2011). Hard real-time computing systems: predictable scheduling algorithms and applications. Vol. 24. Springer Science & Business Media.

Cheng, A. M. K. & Ras, J. (2007). The implementation of the priority ceiling protocol in Ada-2005. ACM SIGAda Ada Letters 27(1), 24-39. https://doi.org/10.1145/1274610.1274611

Jejurikar, R., & Gupta, R. (2002). Energy aware edf scheduling with task synchronization for embedded real time systems. Proc. of the Workshop on Compilers and Operating Systems for Low Power.

Jejurikar, R. & Gupta, R. (2006). Energy-aware task scheduling with task synchronization for embedded real-time systems. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 25(6), 1024-1037.

Laplante, P.A. & Ovaska, S.J. (2012). Real-Time Systems Design and Analysis: Tools for the Practitioner. 4th ed. Hoboken, NJ: JOHN WILEY & SONS.

Lipskoch, H., Albers, K., & Slomka, F. (2007). Fast Calculation of Permissible Slowdown Factors for Hard Real-Time Systems. Proceedings of the 17th international conference on Integrated Circuit and System Design: power and timing modeling, optimization and simulation (PATMOS'07), Sweden, pp.495-504.

Liu, J. W. S., (2000). Real-Time Systems. Upper Saddle River, NJ: Prentice-Hall.

Liu, L. & Layland, J. W. (1973). Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment. J. ACM.

Palamut, S., Gönültaş, T., Elewi, A., & Avaroğlu, E. (2019). Task Scheduling Algorithms and Resource Access Protocols in Real Time Systems. In the Proceedings of 2019 International Artificial Intelligence and Data Processing Symposium (IDAP). https://doi.org/10.1109/IDAP.2019.8875974

Pillai, P. & Shin, K.G. (2001). Real-time dynamic voltage scaling for low-power embedded operating systems. SIGOPS Oper. Syst. Rev. 35(5), 89–102. DOI:https://doi.org/10.1145/502059.502044

Saad, E.M., Elewi, A., Shalan, M. & Awadalla, M. (2013). Energy and synchronization-aware mapping of real-time tasks on asymmetric multicore platforms. International Journal of Computer Applications (IJCA), 75.11 (2013): 35-40. https://doi.org/10.5120/13159-0932

Schmitz, M. T., Al-Hashimi, B. M., & Eles, P. (2005). System-level design techniques for energy-efficient embedded systems. System-Level Design Techniques for Energy-Efficient Embedded Systems (pp. 1–194). Springer US. https://doi.org/10.1007/b106642

Sha, L. & Goodenough, J. (1990). Real-time scheduling theory and Ada. IEEE Transactions on Computer, Vol. 23, No. 4.

Wu, J. & Huang, Y. (2017). MCRTsim: A simulation tool for multi-core real-time systems. In 2017 International Conference on Applied System Innovation (ICASI), 461–464.

Yıldırım, M. K. , Süder, L. , Alkader, A. A. & Elewi, A. (2020). Çok İşlemcili Gerçek Zamanlı Sistemlerde Zamanlama Algoritmaları . Bilgisayar Bilimleri ve Teknolojileri Dergisi , 1 (2) , 42-50 . Retrieved from https://dergipark.org.tr/en/pub/bibted/issue/57253/788353