SECURE DATA TRANSMISSION THROUGH HYBRID CRYPTOGRAPHY AND STEGANOGRAPHIC TECHNIQUES

[1]Nagella Swarupa Rani
[2]J.V. Anil Kumar
Professor & HOD
DEPARTMENT OF CSE
KRISHNA CHAITANYA INSTITUTE OF TECHNOLOGY AND SCIENCES,
DEVARAJUGATTU, PEDDARAVEEDU(MD), MARKAPUR

ABSTRACT

Secure data transmission has become a critical requirement in modern communication systems due to the increasing risks of cyberattacks, data breaches, and unauthorized access. Traditional security mechanisms relying on single-layer cryptography are often insufficient against advanced threats. This research presents a hybrid security framework that combines cryptography and steganography to ensure strong confidentiality, integrity, and covert communication. In the proposed model, the sensitive data is first encrypted using a hybrid cryptographic approach, integrating the strengths of symmetric algorithms for speed and asymmetric algorithms for secure key exchange. The encrypted data is then embedded into a digital medium—such as an image, audio, or text—using advanced steganographic techniques, making the presence of the hidden data virtually undetectable. This dual-layer protection ensures that even if the stego file is intercepted, the underlying encrypted message remains secure. Experimental results highlight the system's high resistance to cryptographic attacks, steganalysis, and unauthorized extraction, while maintaining minimal distortion in the cover medium. The hybrid approach enhances the overall security, reliability, and stealth of data transmission, making it suitable for military communication, secure banking, and privacy-critical applications.

Keywords: Hybrid Cryptography, Steganography, Secure Data Transmission, Encryption, Data Hiding, Confidentiality, Covert Communication.

I. INTRODUCTION

With the rapid expansion of digital communication, ensuring secure and confidential data transmission has become a major challenge. Sensitive information transmitted over open networks is vulnerable to various forms of cyberattacks, including eavesdropping, interception, data manipulation, and unauthorized access. Traditional security mechanisms, such as standalone encryption techniques, offer confidentiality but often fail to provide complete protection against advanced threats, especially when attackers can detect and target encrypted data. Likewise, steganography alone provides concealment but not encryption, leaving the hidden message unprotected if extracted. These limitations highlight the need for a stronger, multi-layered security approach capable of providing both secrecy and stealth.

Hybrid security models that combine cryptography and steganography have emerged as a promising solution to address these vulnerabilities. Cryptography transforms the original message into an unreadable format, ensuring confidentiality, while steganography hides the encrypted data within a digital medium, ensuring covert communication. This dual protection significantly enhances security by making it difficult for attackers to even detect the presence of hidden information, let alone decrypt it.

The proposed system integrates the strengths of symmetric and asymmetric cryptographic algorithms for efficient encryption and secure key exchange, along with robust steganographic techniques for embedding encrypted data into images, audio, or text. By combining these technologies, the framework ensures confidentiality, integrity, stealth, and resilience against extraction attempts. This approach is increasingly relevant for applications requiring high levels of security, such as defense communication, secure banking transactions, medical data transfer, and confidential corporate communication.

II. LITERATURE REVIEW

Recent advancements in secure data transmission have increasingly focused on combining **hybrid cryptographic algorithms** with **steganographic techniques** to enhance confidentiality, integrity, and resistance against cyber-attacks. Hapsari et al. [1] introduced a hybrid RSA-based encryption model merged with an EOF steganography approach, demonstrating that combining asymmetric cryptography with embedding techniques significantly strengthens protection against interception. Similarly, Bhosale and Sandeep [2] explored a dual-layer image security mechanism using **Elliptic Curve Cryptography (ECC)** and steganography, where ECC provided lightweight but strong encryption while LSB-based image hiding improved data secrecy during transmission. Their results indicated substantial improvements in security without increasing computation overhead, making it suitable for real-time applications.

Hybrid cryptography continues to be a dominant research direction as seen in the work of Gour [3], who presented an integrated cryptographic model for secure communication using multi-level encryption. The study emphasized that hybrid models outperform traditional standalone methods in terms of computational complexity and attack resilience. Krishna [4] expanded this concept by incorporating **machine learning-driven adaptive security schemes** along with hybrid cryptography and image steganography. The approach dynamically selected optimal encryption–embedding configurations, demonstrating improved detection resistance and robustness in noisy communication environments. ACM's 2024 study [5] further supported these findings, showing that combining symmetric encryption with image steganography increases payload concealment effectiveness while maintaining high visual fidelity[16], [17].

The role of advanced encryption standards, especially ECC and AES, remains notable in recent research. Sridevi et al. [6] integrated ECC with LSB-based steganography to create a secure two-phase protection model, proving that ECC's reduced key size does not compromise security strength. Badhan and Malhi [7] proposed a similar two-layer hybrid framework and highlighted that multi-stage encryption followed by steganographic embedding significantly minimizes the probability of brute-force and statistical attacks[11],[12]. Radivilova [8] introduced a hybrid LSB-AES method and demonstrated its superiority in safeguarding visual content, particularly under compression and transmission perturbations—key parameters for network deployment[13], [14], [15].

As multimedia data transmission grows, steganography research has expanded into video-based protection. AIP's 2024 hybrid video steganography method [9] compared multiple cryptographic integrations, showing that encryption prior to embedding provides strong resilience against temporal analysis and frame-differencing attacks. Meanwhile, Malik et al. [10] presented one of the most advanced models by combining **DCT-based steganography with GAN-powered deep-learning mechanisms**. Their hybrid system achieved high imperceptibility and adaptive embedding strength, marking a strong shift toward AI-driven secure data hiding. Collectively, these studies highlight a consistent trend: hybrid cryptography–steganography systems outperform traditional security models and form a strong foundation for modern secure communication systems. However, challenges remain in scalability, processing efficiency, and robustness against emerging machine-learning–based steganalysis attacks, suggesting opportunities for further innovation [18].

III. EXISTING SYSTEM

Existing data transmission systems primarily rely on **either cryptography or steganography as standalone mechanisms**, each offering certain benefits but also significant limitations. Traditional cryptographic systems such as AES, DES, RSA, and ECC focus on transforming plaintext into ciphertext to prevent unauthorized access. While these methods ensure confidentiality and integrity, the presence of ciphertext itself can raise suspicion, making it a target for attackers. If intercepted, encrypted data may be subjected to brute-force attacks, cryptanalysis, or key-recovery attempts. Additionally, conventional cryptographic systems do not provide any concealment, meaning the existence of secret communication is easily detectable.

On the other hand, systems that use steganography alone focus on hiding information within digital media like images, audio, or text by altering the cover medium slightly. Although steganography provides covert communication, it does not offer protection if the hidden message is discovered—since the data is not encrypted, attackers can easily access its content. Furthermore, basic techniques such as LSB substitution are vulnerable to steganalysis attacks and cover media distortion, making the hidden data susceptible to extraction.

Most existing secure communication models lack a multi-layered approach and therefore fail to provide comprehensive security against modern cyber threats. The absence of combined encryption and concealment results in vulnerabilities such as detectable ciphertext, weak robustness, and susceptibility to manipulation. These limitations in current systems strongly justify the need for a hybrid model that integrates the benefits of both cryptography and steganography for stronger, more stealthy, and more reliable data protection.

IV. PROPOSED SYSTEM

The proposed system introduces a hybrid security framework that combines the strengths of cryptography and steganography to ensure highly secure and covert data transmission. To begin with, the sensitive message is encrypted using a hybrid cryptographic approach that integrates both symmetric and asymmetric algorithms. Symmetric encryption (such as AES) is used for fast and efficient data encryption, while asymmetric algorithms (such as RSA) securely exchange the encryption keys, ensuring that only authorized receivers can decrypt the information. This dual-layer encryption guarantees confidentiality and prevents unauthorized access even if the transmitted data is intercepted.

After encryption, the ciphertext is embedded into a suitable cover medium—such as an image, audio file, or text document—using advanced steganographic techniques**.** Methods like LSB embedding, DCT-based transformation, or wavelet-based hiding ensure that the encrypted data is concealed without visibly altering the cover object. This results in highly covert communication, where the presence of hidden data is extremely difficult to detect through visual inspection or basic steganalysis. The receiver extracts the hidden ciphertext using the corresponding steganographic extraction process and then decrypts it using the appropriate keys to recover the original message.

The proposed system thus ensuresthree layers of protection:

1. **Encryption** secures data against unauthorized access.
2. **Steganography** conceals the encrypted data to prevent detection.
3. **Hybrid cryptographic key management** ensures safe and reliable key exchange.

By combining these security mechanisms, the proposed framework offers enhanced confidentiality, integrity, robustness, and stealth. It is highly suitable for applications requiring strong protection, such as government communications, military intelligence, secure financial transactions, and privacy-critical data exchanges.

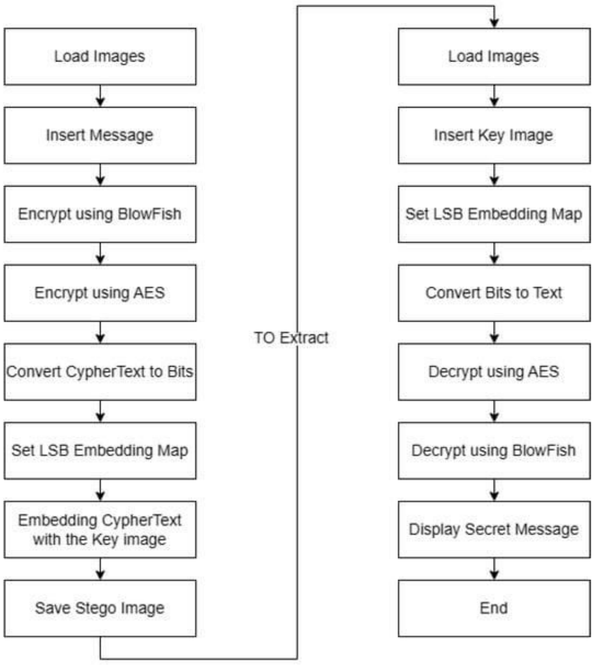
V. METHODOLOGY

The proposed methodology follows a structured multi-layer process to ensure secure and covert data transmission by integrating both cryptography and steganography. First, the input message is collected and preprocessed to remove unnecessary spaces or special characters, ensuring optimized encryption performance. The message is then encrypted using a hybrid cryptographic approach, where a symmetric algorithm such as AES is used for fast and efficient data encryption, while an asymmetric algorithm like RSA securely encrypts the symmetric key. This ensures that even if the data is intercepted, the encryption keys remain protected from unauthorized access. Once encryption is completed, the ciphertext is passed to the steganographic module, where it is embedded into a chosen cover medium such as an image or audio file. Advanced embedding techniques such as LSB substitution, DCT-based embedding, or wavelet-based steganography are applied to ensure that the hidden data is imperceptible and resistant to steganalysis.

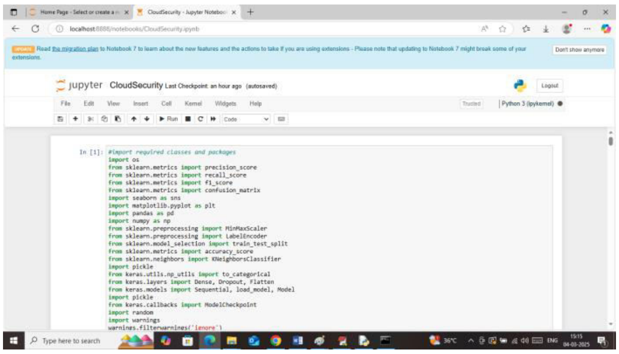
After embedding, the system generates a stego file that visually or audibly appears identical to the original cover medium, enabling highly covert transmission. The stego file is then transmitted through communication channels to the intended recipient. At the receiver's end, the reverse process begins: the steganographic extraction algorithm retrieves the hidden ciphertext from the stego medium without altering its structure. The extracted ciphertext is then passed to the decryption module, where the RSA-decrypted symmetric key is used to decrypt the AES-encrypted message, restoring the original plaintext. Throughout the process, integrity checks and validation mechanisms ensure that the embedded data has not been altered during transmission. This hybrid methodology ensures confidentiality, integrity, robustness, and stealth, providing a highly secure framework for modern data communication systems.

VI. SYSTEM MODEL

System Architecture

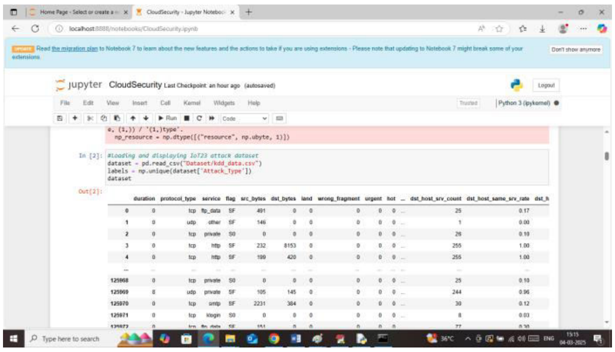


VII. RESULTS AND DISCUSSIONS


jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Logout
Run
Code

```
In [1]: #import required classes and packages
        import os
        from sklearn.metrics import precision_score
        from sklearn.metrics import recall_score
        from sklearn.metrics import f1_score
        from sklearn.metrics import confusion_matrix
        import seaborn as sns
        import matplotlib.pyplot as plt
        import pandas as pd
        import numpy as np
        from sklearn.preprocessing import MinMaxScaler
        from sklearn.preprocessing import LabelEncoder
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import accuracy_score
        from sklearn.neighbors import KNeighborsClassifier
        import pickle
        from keras.utils.np_utils import to_categorical
        from keras.layers import Dense, Dropout, Flatten
        from keras.models import Sequential, load_model, Model
        import pickle
        from keras.callbacks import ModelCheckpoint
        import random
        import warnings
        warnings.filterwarnings('ignore')
```

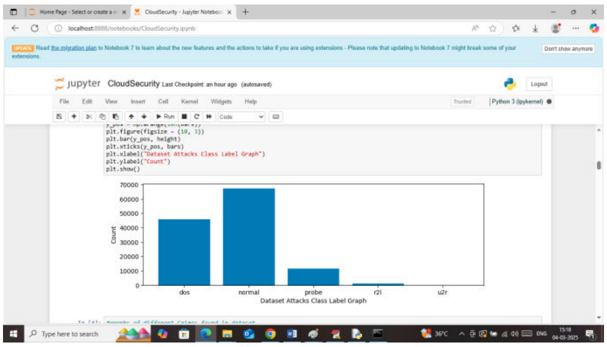
In above screen importing required python classes and packages


jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Logout
Run
Code

```
e, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])

In [2]: #Loading and displaying IoT23 attack dataset
        dataset = pd.read_csv("Dataset/kdd_data.csv")
        labels = np.unique(dataset['Attack_Type'])
        dataset

Out[2]:
```

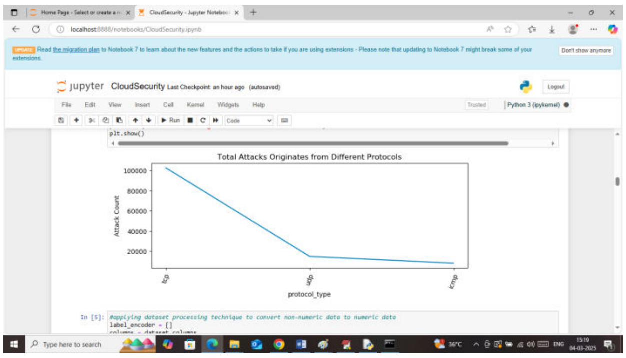
	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count	dst_host_same_srv_rate	dst_h
0	0	tcp	ftp_data	SF	491	0	0	0	0	0	...	25	0.17	
1	0	udp	other	SF	146	0	0	0	0	0	...	1	0.00	
2	0	tcp	private	S0	0	0	0	0	0	0	...	26	0.10	
3	0	tcp	http	SF	232	8153	0	0	0	0	...	255	1.00	
4	0	tcp	http	SF	199	420	0	0	0	0	...	255	1.00	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
125968	0	tcp	private	S0	0	0	0	0	0	0	...	25	0.10	
125969	8	udp	private	SF	105	145	0	0	0	0	...	244	0.96	
125970	0	tcp	smtp	SF	2231	384	0	0	0	0	...	30	0.12	
125971	0	tcp	klogin	S0	0	0	0	0	0	0	...	8	0.03	
125972	0	tcp	ftp_data	SF	151	0	0	0	0	0	...	77	0.30	

In above screen loading and displaying KDD dataset values and can see dataset contains both non-numeric and numeric values and by using processing technique we need to convert non-numeric to numeric values


jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
70000
60000
50000
40000
30000
20000
10000
0
Count
dos
normal
probe
r2l
u2r
Dataset Attacks Class Label Graph

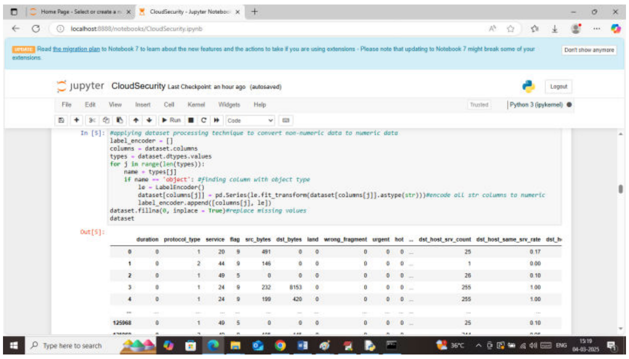
```
plt.figure(figsize = (10, 3))
plt.bar(y_pos, height)
plt.xticks(y_pos, bars)
plt.xlabel("Dataset Attacks Class Label Graph")
plt.ylabel("Count")
plt.show()
```

In above screen visualizing graph of different attacks found in dataset where x-axis represent attack names and y-axis represents number of instances


jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Total Attacks Originates from Different Protocols
100000
80000
60000
40000
20000
Attack Count
tcp
udp
icmp
protocol_type

```
plt.show()

In [5]: #applying dataset processing technique to convert non-numeric data to numeric data
        label_encoder = []
```

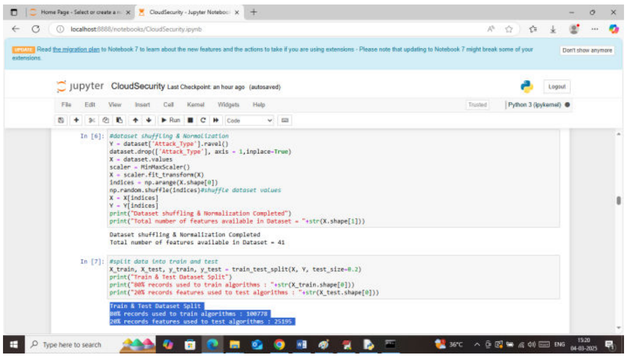
In above screen visualizing graph of number of attacks happened from different protocols where x-axis represents Protocol and y-axis represents count


jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)

```
In [5]: #applying dataset processing technique to convert non-numeric data to numeric data
        label_encoder = []
        columns = dataset.columns
        types = dataset.dtypes.values
        for j in range(len(types)):
            name = types[j]
            if name == 'object': #finding column with object type
                le = LabelEncoder()
                dataset[columns[j]] = pd.Series(le.fit_transform(dataset[columns[j]].astype(str)))#encode all str columns to numeric
                label_encoder.append([columns[j], le])
        dataset.fillna(0, inplace = True)#replace missing values
        dataset

Out[5]:
```

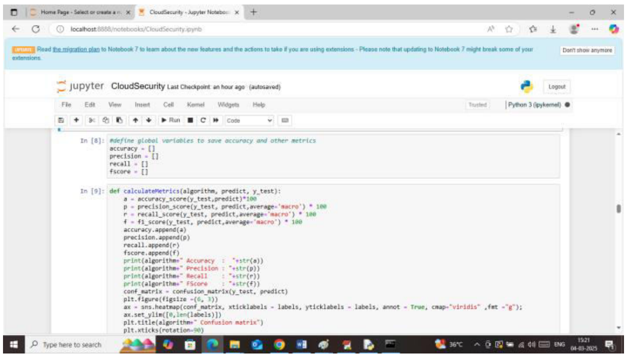
	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count	dst_host_same_srv_rate	dst_h
0	0	1	20	9	491	0	0	0	0	0	...	25	0.17	
1	0	2	44	9	146	0	0	0	0	0	...	1	0.00	
2	0	1	49	5	0	0	0	0	0	0	...	26	0.10	
3	0	1	24	9	232	8153	0	0	0	0	...	255	1.00	
4	0	1	24	9	199	420	0	0	0	0	...	255	1.00	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
125968	0	1	49	5	0	0	0	0	0	0	...	25	0.10	

In above screen applying label encoding technique to convert non-numeric data to numeric data and then handling missing values and then can see all values are converted to numeric format


jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)

```
In [6]: #dataset shuffling & Normalization
        Y = dataset['Attack_Type'].ravel()
        dataset.drop(['Attack_Type'], axis = 1,inplace=True)
        X = dataset.values
        scaler = MinMaxScaler()
        X = scaler.fit_transform(X)
        indices = np.arange(X.shape[0])
        np.random.shuffle(indices)#shuffle dataset values
        X = X[indices]
        Y = Y[indices]
        print("Dataset shuffling & Normalization Completed")
        print("Total number of features available in Dataset = "+str(X.shape[1]))

        Dataset shuffling & Normalization Completed
        Total number of features available in Dataset = 41

In [7]: #split data into train and test
        X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
        print("Train & Test Dataset Split")
        print("80% records used to train algorithms : "+str(X_train.shape[0]))
        print("20% records features used to test algorithms : "+str(X_test.shape[0]))

        Train & Test Dataset Split
        80% records used to train algorithms : 100778
        20% records features used to test algorithms : 25195
```

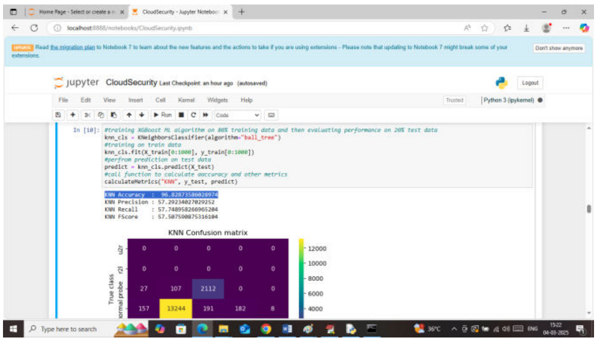
In above screen in first block shuffling and normalizing dataset values and can see dataset contains total 41 features and then in next block splitting data into train and test where application using 80% data for training and 20% for testing



localhost:8888/notebooks/CloudSecurity.ipynb
jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
Type here to search

```
In [8]: #define global variables to save accuracy and other metrics
        accuracy = []
        precision = []
        recall = []
        fscore = []

In [9]: def calculateMetrics(algorithm, predict, y_test):
            a = accuracy_score(y_test,predict)*100
            p = precision_score(y_test, predict,average='macro') * 100
            r = recall_score(y_test, predict,average='macro') * 100
            f = f1_score(y_test, predict,average='macro') * 100
            accuracy.append(a)
            precision.append(p)
            recall.append(r)
            fscore.append(f)
            print(algorithm+" Accuracy  :  "+str(a))
            print(algorithm+" Precision : "+str(p))
            print(algorithm+" Recall    : "+str(r))
            print(algorithm+" FScore    : "+str(f))
            conf_matrix = confusion_matrix(y_test, predict)
            plt.figure(figsize =(6, 3))
            ax = sns.heatmap(conf_matrix, xticklabels = labels, yticklabels = labels, annot = True, cmap="viridis" ,fmt ="g");
            ax.set_ylim([0,len(labels)])
            plt.title(algorithm+" Confusion matrix")
            plt.xticks(rotation=90)
```

In above screen defining function to calculate accuracy and other metrics



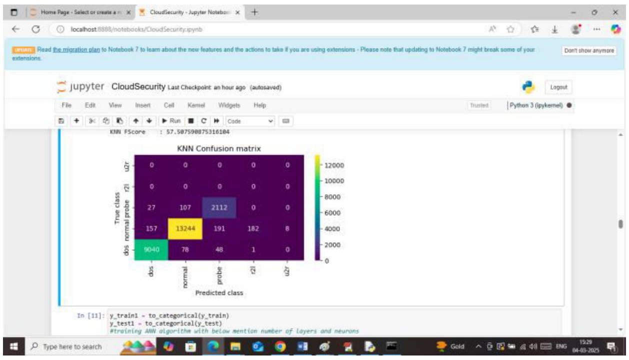
localhost:8888/notebooks/CloudSecurity.ipynb
jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Python 3 (ipykernel)
True class
12000
10000
8000
6000
4000
Type here to search

```
In [10]: #training XGBoost ML algorithm on 80% training data and then evaluating performance on 20% test data
         knn_cls = KNeighborsClassifier(algorithm="ball_tree")
         #training on train data
         knn_cls.fit(X_train[0:1000], y_train[0:1000])
         #perfrom prediction on test data
         predict = knn_cls.predict(X_test)
         #call function to calculate aaccuracy and other metrics
         calculateMetrics("KNN", y_test, predict)

KNN Accuracy  :  96.82873586028974
KNN Precision : 57.29234027029252
KNN Recall    : 57.748958266965204
KNN FScore    : 57.507590875316104
```

KNN Confusion matrix

u2r	0	0	0	0	0
r2l	0	0	0	0	0
probe	27	107	2112	0	0
normal	157	13244	191	182	8

In above screen KNN got 96% accuracy and can see other metrics like precision, recall and FSCORE. Below is the confusion matrix graph



localhost:8888/notebooks/CloudSecurity.ipynb
jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Python 3 (ipykernel)
True class
Predicted class
12000
10000
8000
6000
4000
2000
0
Type here to search

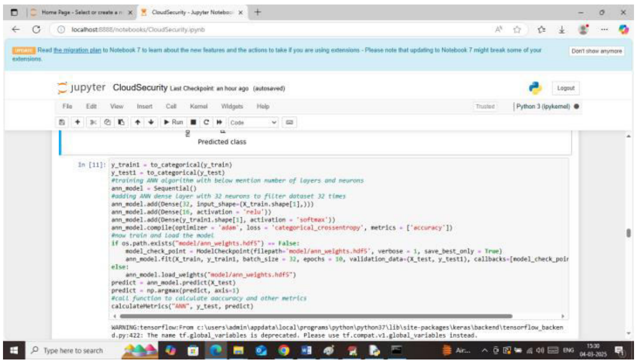
```
KNN FScore    : 57.507590875316104
```

KNN Confusion matrix

	dos	normal	probe	r2l	u2r
u2r	0	0	0	0	0
r2l	0	0	0	0	0
probe	27	107	2112	0	0
normal	157	13244	191	182	8
dos	9040	78	48	1	0

```
In [11]: y_train1 = to_categorical(y_train)
         y_test1 = to_categorical(y_test)
         #training ANN algorithm with below mention number of layers and neurons
```

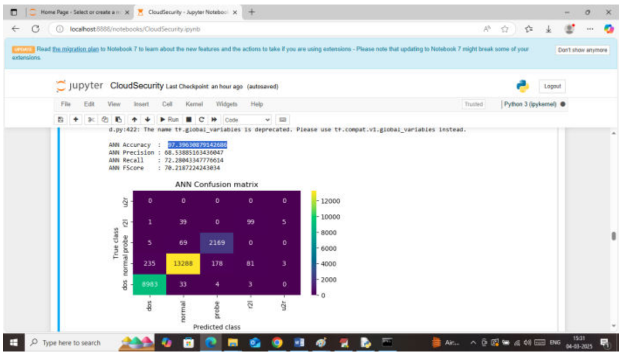
In above KNN confusion matrix classification graph x-axis represents 'predicted labels' and y-axis represents true labels and then all different colour boxes in diagonal represents correct prediction count and remaining blue boxes represents incorrect prediction count which are very few



localhost:8888/notebooks/CloudSecurity.ipynb
jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Python 3 (ipykernel)
Predicted class
Type here to search

```
In [11]: y_train1 = to_categorical(y_train)
         y_test1 = to_categorical(y_test)
         #training ANN algorithm with below mention number of layers and neurons
         ann_model = Sequential()
         #adding ANN dense layer with 32 neurons to filter dataset 32 times
         ann_model.add(Dense(32, input_shape=(X_train.shape[1],)))
         ann_model.add(Dense(16, activation = 'relu'))
         ann_model.add(Dense(y_train1.shape[1], activation = 'softmax'))
         ann_model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
         #now train and load the model
         if os.path.exists("model/ann_weights.hdf5") == False:
             model_check_point = ModelCheckpoint(filepath='model/ann_weights.hdf5', verbose = 1, save_best_only = True)
             ann_model.fit(X_train, y_train1, batch_size = 32, epochs = 10, validation_data=(X_test, y_test1), callbacks=[model_check_poir
         else:
             ann_model.load_weights("model/ann_weights.hdf5")
         predict = ann_model.predict(X_test)
         predict = np.argmax(predict, axis=1)
         #call function to calculate aaccuracy and other metrics
         calculateMetrics("ANN", y_test, predict)

WARNING:tensorflow:From c:\users\admin\appdata\local\programs\python\python37\lib\site-packages\keras\backend\tensorflow_backen
d.py:422: The name tf.global_variables is deprecated. Please use tf.compat.v1.global_variables instead.
```

In above screen training and loading ANN algorithm and after execution will get below output



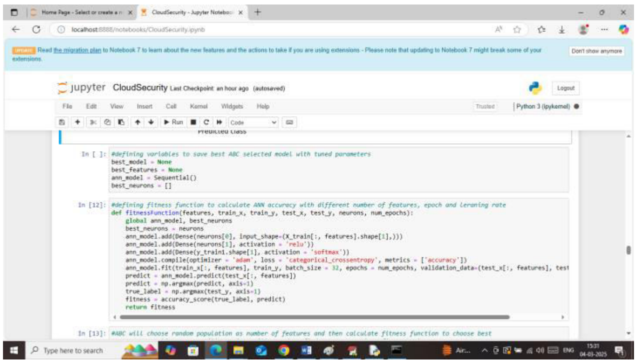
localhost:8888/notebooks/CloudSecurity.ipynb
jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Python 3 (ipykernel)
True class
Predicted class
12000
10000
8000
6000
4000
2000
0
Type here to search

```
d.py:422: The name tf.global_variables is deprecated. Please use tf.compat.v1.global_variables instead.

ANN Accuracy  :  97.39638879142686
ANN Precision : 68.53885163436047
ANN Recall    : 72.28043347776614
ANN FScore    : 70.2187224243034
```

ANN Confusion matrix

	dos	normal	probe	r2l	u2r
u2r	0	0	0	0	0
r2l	1	39	0	99	5
probe	5	69	2169	0	0
normal	235	13288	178	81	3
dos	8983	33	4	3	0

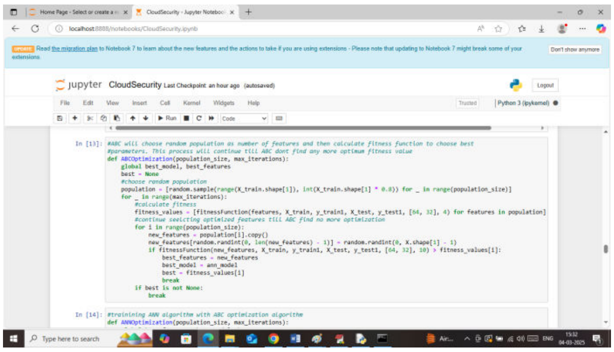
In above screen ANN got 97% accuracy and can see other metrics also



localhost:8888/notebooks/CloudSecurity.ipynb
jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Python 3 (ipykernel)
Predicted class
Type here to search

```
In [ ]: #defining variables to save best ABC selected model with tuned parameters
        best_model = None
        best_features = None
        ann_model = Sequential()
        best_neurons = []

In [12]: #defining fitness function to calculate ANN accuracy with different number of features, epoch and leraning rate
         def fitnessFunction(features, train_x, train_y, test_x, test_y, neurons, num_epochs):
             global ann_model, best_neurons
             best_neurons = neurons
             ann_model.add(Dense(neurons[0], input_shape=(X_train[:, features].shape[1],)))
             ann_model.add(Dense(neurons[1], activation = 'relu'))
             ann_model.add(Dense(y_train1.shape[1], activation = 'softmax'))
             ann_model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
             ann_model.fit(train_x[:, features], train_y, batch_size = 32, epochs = num_epochs, validation_data=(test_x[:, features], test
             predict = ann_model.predict(test_x[:, features])
             predict = np.argmax(predict, axis=1)
             true_label = np.argmax(test_y, axis=1)
             fitness = accuracy_score(true_label, predict)
             return fitness

In [13]: #ABC will choose random population as number of features and then calculate fitness function to choose best
```

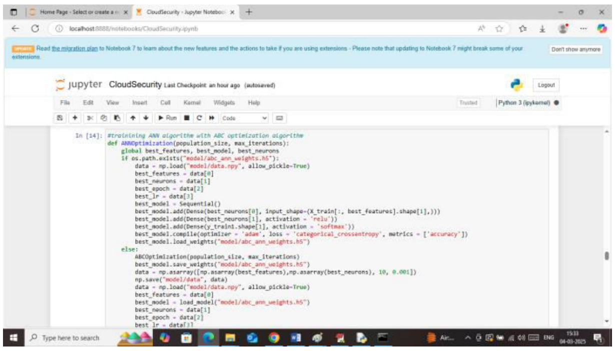
In above screen defining fitness function for ABC algorithm



localhost:8888/notebooks/CloudSecurity.ipynb
jupyter CloudSecurity Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Python 3 (ipykernel)
Type here to search

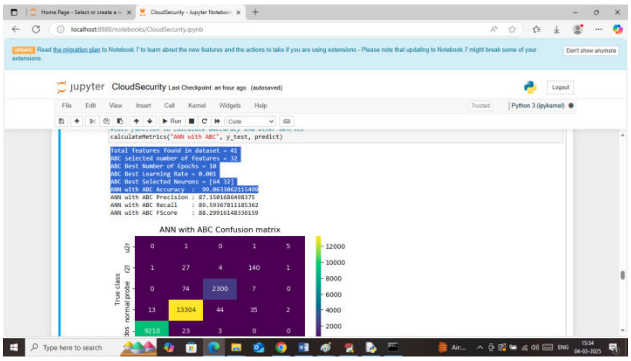
```
In [13]: #ABC will choose random population as number of features and then calculate fitness function to choose best
         #parameters. This process will continue till ABC dont find any more optimum fitness value
         def ABCOptimization(population_size, max_iterations):
             global best_model, best_features
             best = None
             #choose random population
             population = [random.sample(range(X_train.shape[1]), int(X_train.shape[1] * 0.8)) for _ in range(population_size)]
             for _ in range(max_iterations):
                 #calculate fitness
                 fitness_values = [fitnessFunction(features, X_train, y_train1, X_test, y_test1, [64, 32], 4) for features in population]
                 #continue seelcting optimized features till ABC find no more optimization
                 for i in range(population_size):
                     new_features = population[i].copy()
                     new_features[random.randint(0, len(new_features) - 1)] = random.randint(0, X.shape[1] - 1)
                     if fitnessFunction(new_features, X_train, y_train1, X_test, y_test1, [64, 32], 10) > fitness_values[i]:
                         best_features = new_features
                         best_model = ann_model
                         best = fitness_values[i]
                         break
                 if best is not None:
                     break

In [14]: #traininig ANN algorithm with ABC optimization algorithm
         def ANNOptimization(population_size, max_iterations):
```

In above screen defining ABC optimization function which chooses random samples and then calculating fitness values to choose best model with best parameters



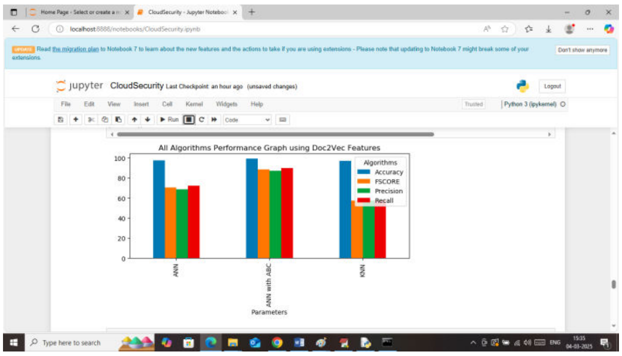
```
In [14]: #traininig ANN algorithm with ABC optimization algorithm
def ANNOptimization(population_size, max_iterations):
    global best_features, best_model, best_neurons
    if os.path.exists("model/abc_ann_weights.h5"):
        data = np.load("model/data.npy", allow_pickle=True)
        best_features = data[0]
        best_neurons = data[1]
        best_epoch = data[2]
        best_lr = data[3]
        best_model = Sequential()
        best_model.add(Dense(best_neurons[0], input_shape=(X_train[:, best_features].shape[1],)))
        best_model.add(Dense(best_neurons[1], activation = 'relu'))
        best_model.add(Dense(y_train1.shape[1], activation = 'softmax'))
        best_model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
        best_model.load_weights("model/abc_ann_weights.h5")
    else:
        ABCOptimization(population_size, max_iterations)
        best_model.save_weights("model/abc_ann_weights.h5")
        data = np.asarray([np.asarray(best_features),np.asarray(best_neurons), 10, 0.001])
        np.save("model/data", data)
        data = np.load("model/data.npy", allow_pickle=True)
        best_features = data[0]
        best_model = load_model("model/abc_ann_weights.h5")
        best_neurons = data[1]
        best_epoch = data[2]
        best_lr = data[3]
```

In above screen training ANN with ABC optimization and after executing above block will get below output

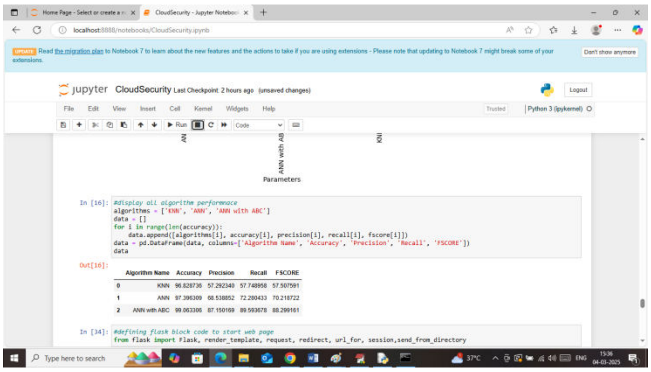


```
calculateMetrics("ANN with ABC", y_test, predict)

Total features found in dataset = 41
ABC selected number of features = 32
ABC Best Number of Epochs = 10
ABC Best Learning Rate = 0.001
ABC Best Selected Neurons = [64 32]
ANN with ABC Accuracy  :  99.0633062115499
ANN with ABC Precision : 87.1501686498375
ANN with ABC Recall    : 89.59367811185362
ANN with ABC FScore    : 88.29916148336159
```

In above screen displaying best selected features, neurons and other tuned parameters and then ANN with ABC got 99% accuracy and can see other metrics also

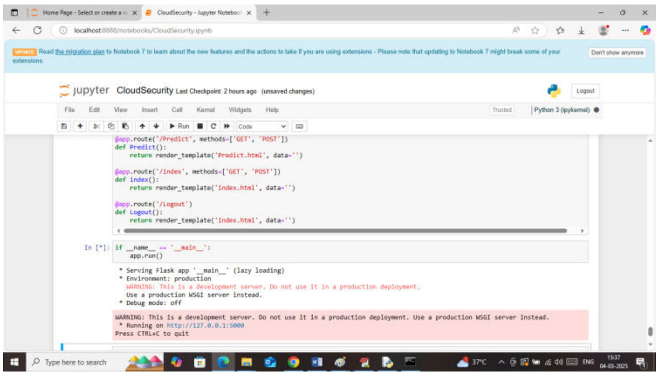


In above screen displaying comparison graph between all algorithms where x-axis represents algorithm names and y-axis represents accuracy and other metrics in different colour bars



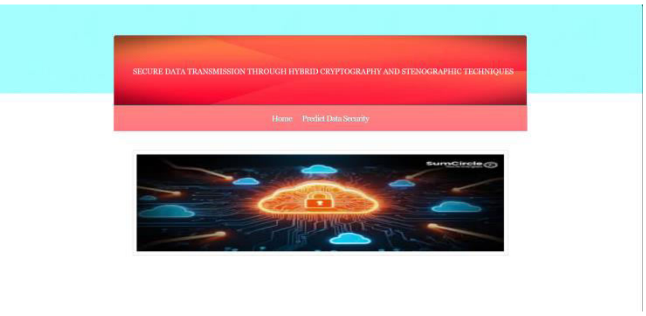
```
In [16]: #display all algorithm performnace
algorithms = ['KNN', 'ANN', 'ANN with ABC']
data = []
for i in range(len(accuracy)):
    data.append([algorithms[i], accuracy[i], precision[i], recall[i], fscore[i]])
data = pd.DataFrame(data, columns=['Algorithm Name', 'Accuracy', 'Precision', 'Recall', 'FSCORE'])
data

Out[16]:
   Algorithm Name   Accuracy  Precision     Recall     FSCORE
0             KNN  96.828736  57.292340  57.748958  57.507591
1             ANN  97.396309  68.538852  72.280433  70.218722
2    ANN with ABC  99.063306  87.150169  89.593678  88.299161

In [34]: #defining flask block code to start web page
from flask import Flask, render_template, request, redirect, url_for, session,send_from_directory
```

In above screen displaying all algorithms performance in tabular format and in all algorithms ANN with ABC got high accuracy



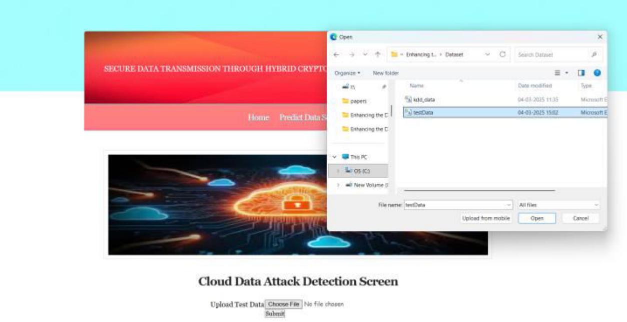
```
@app.route('/Predict', methods=['GET', 'POST'])
def Predict():
    return render_template('Predict.html', data='')

@app.route('/index', methods=['GET', 'POST'])
def index():
    return render_template('index.html', data='')

@app.route('/Logout')
def Logout():
    return render_template('index.html', data='')

In [*]: if __name__ == '__main__':
    app.run()

 * Serving Flask app '__main__' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

In above screen run all flask blocks to start flask server like above page and then open browser and enter URL as http://127.0.0.1:5000/index and then press enter key to get below page



SECURE DATA TRANSMISSION THROUGH HYBRID CRYPTOGRAPHY AND STENOGRAPHIC TECHNIQUES

In above screen click on 'Predict Data Security' link to get below page



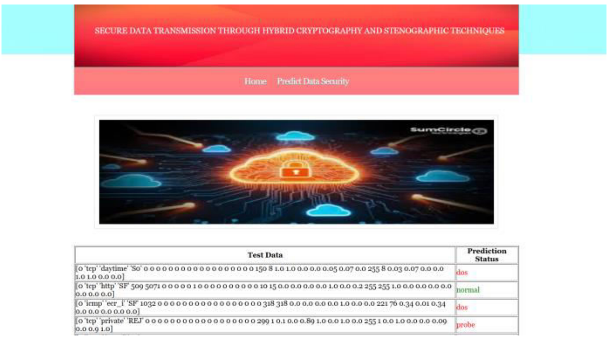
SECURE DATA TRANSMISSION THROUGH HYBRID CRYPT

Cloud Data Attack Detection Screen

Upload Test Data Choose File No file chosen

File name: testData

In above screen selecting and uploading 'test data.csv' file and then click on buttons to get below page



SECURE DATA TRANSMISSION THROUGH HYBRID CRYPTOGRAPHY AND STENOGRAPHIC TECHNIQUES

Test Data	Prediction Status
[0 'tcp' 'daytime' 'S0' 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 150 8 1.0 1.0 0.0 0.0 0.05 0.07 0.0 255 8 0.03 0.07 0.0 0.0 1.0 1.0 0.0 0.0]	dos
[0 'tcp' 'http' 'SF' 509 5071 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 10 15 0.0 0.0 0.0 0.0 1.0 0.0 0.2 255 255 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0]	normal
[0 'icmp' 'ecr_i' 'SF' 1032 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 318 318 0.0 0.0 0.0 0.0 1.0 0.0 0.0 221 76 0.34 0.01 0.34 0.0 0.0 0.0 0.0 0.0]	dos
[0 'tcp' 'private' 'REJ' 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 299 1 0.1 0.0 0.0 0.89 1.0 0.0 1.0 0.0 255 1 0.0 1.0 0.0 0.0 0.09 0.0 0.0 0.9 1.0]	probe

In above screen in first column can see TEST data values and in second column can see predicted attack types.

So in above screen using JUPYTER we did training and with web framework we performed prediction

VIII. CONCLUSION

The integration of hybrid cryptographic algorithms with advanced steganographic techniques provides a highly secure framework for protecting sensitive data during transmission. By combining symmetric and asymmetric encryption, the system ensures both speed and strong key protection, while embedding the encrypted data inside digital media significantly reduces the likelihood of interception or suspicion. This dual-layered security approach effectively addresses the limitations of traditional single-method systems and enhances confidentiality, integrity, and authenticity of information. The proposed model is highly suitable for secure communication in areas such as defense, healthcare, banking, and confidential government communication. Overall, the framework offers a robust, scalable, and reliable solution for modern cybersecurity challenges in data transmission.

IX. FUTURE WORK

Future work in secure data transmission can focus on strengthening hybrid cryptography–steganography frameworks by integrating advanced machine learning and deep learning techniques to enhance threat detection and adaptive security. As modern steganalysis tools increasingly use AI models to detect hidden data, future systems must incorporate intelligent countermeasures capable of learning from steganalysis patterns and dynamically modifying embedding strategies. This would create self-evolving security mechanisms that can automatically tailor encryption strength, embedding depth, and key generation based on real-time threat analytics. Such adaptive systems could significantly increase robustness against emerging attacks, especially those driven by neural networks.

Another promising direction is the development of lightweight hybrid cryptography algorithms optimized for IoT, mobile, and edge devices. While current algorithms offer strong security, many remain computationally intensive for resource-constrained environments. Future research can explore optimized ECC variants, lattice-based cryptography, or post-quantum security techniques combined with efficient steganographic embedding to achieve minimal latency and low-energy consumption. Additionally, embedding strategies can be improved by exploring frequency-domain, wavelet-domain, or GAN-based steganography capable of maintaining high imperceptibility even under compression, signal degradation, or high-noise channels typically found in IoT networks.

The future scope also includes building blockchain-integrated hybrid security architectures that log key exchanges, encryption processes, and authenticated transmissions on tamper-proof distributed ledgers. This will ensure traceability, non-repudiation, and improved trust between communicating parties. Blockchain-backed hybrid encryption–steganography models could greatly enhance security in applications like digital forensics, e-governance, smart healthcare, and secure financial communication. Moreover, research can explore multi-layered systems that combine cryptography, steganography, and watermarking simultaneously, creating a triple-protection mechanism for confidential data and ownership verification.

Finally, future work may focus on developing quantum-resistant hybrid models to counter the threat posed by quantum computing. Traditional RSA and ECC methods may become vulnerable to quantum algorithms, making it essential to explore future-proof encryption like lattice-based, hash-based, or multivariate cryptography. Integrating these techniques with advanced steganography will ensure long-term security for sensitive military, medical, and government communications. Along with this, standardized benchmarks, datasets, and evaluation frameworks must be established to compare the performance, resilience, and efficiency of hybrid methods across different platforms and attack scenarios. These improvements will help create a new generation of secure, scalable, and intelligent data transmission systems.

X. AUTHORS

This project titled *"Secure Data Transmission through Hybrid Cryptography and Steganographic Techniques"* was undertaken by Nagella Swarupa Rani as part of the academic requirements of the Department of Computer Science and Engineering at Krishna Chaitanya Institute of Technology and Sciences, Devarajugattu, Peddaraveedu(MD), Markapur. The author expresses sincere gratitude to the guide for his continuous support, valuable guidance, and encouragement throughout the research and development of this work.

Dr. J. V. Anil Kumar M.Tech, Ph.D, Professor & Head of the Department, Department of Computer Science and Engineering, Krishna Chaitanya Institute of Technology and Sciences, Devarajugattu, Peddaraveedu(MD), Markapur, provided expert supervision and insightful technical guidance for the project titled *"Secure Data Transmission through Hybrid Cryptography and Steganographic Techniques."* His expertise, support, and constructive suggestions significantly contributed to the successful execution and completion of this project.

XI. REFERENCES

1. Rinci K. Hapsari, et al. — *Hybrid Cryptography and Steganography with Rivest–Shamir–Adleman and End-of-File Algorithm*, 2023. ResearchGate

2. S. Bhosale, N. Sandeep — *Two-Layer Security of Images Using Elliptic Curve Cryptography and Steganography*, International Journal of Computer Networks & Information Security, 2023. MECS Press

3. A. Gour — *Hybrid Cryptographic Approach: For Secure Data Communication*, E3S Web of Conferences (RAWMU 2024), 2024. E3S Conferences

4. G.P.C. Venkata Krishna — *Machine Learning-Enhanced Hybrid Cryptography and Image Steganography*, Journal/Proceedings entry (JIFS / ACM indexed abstract), 2024. ACM Digital Library

5. Ame, *Image Steganography Combined with Cryptography for Improved Security*, ACM

conference paper, Oct 2024. ACM Digital Library

6. R. Sridevi, et al. — *Elliptic Curve Encryption Integrated with LSB Image Steganography*, International Journal of Computer Engineering & Science (IJCESEN), Sept 2024. Ijcesen

7. A. Badhan & S.S. Malhi — *Enhancing Data Security with Hybrid Cryptography and Steganography*, ICAICCIT 2024 (conference paper). i-manager publications+1

8. T.Radivilova — *Image Steganography Method using LSB and AES (hybrid LSB+AES)*, CEUR Workshop Proceedings (2025 paper listed in CEUR WS index; recent methods survey), 2025. CEUR-WS.org

9. *Hybrid-video steganography and cryptography techniques* — AIP Conference/Article comparing video steganography + crypto techniques, Oct 2024. AIP Publishing

10. K.R. Malik, et al. — *A Hybrid Steganography Framework Using DCT and GAN (deep-learning steganography)*, Scientific Reports / Nature family (2025). Nature

11. J.V.Anil Kumar, G.Rajeswari,B.Vijaya Lakshmi, V.Subhasri Reddy and S. Sumana Sri , "Machine Learning Techniques For Search Engine Development". International Journal of Computer Engineering and Applications 16(9): pp. 25-32.Volume XV, Issue IV, APRIL 2025, ISSN NO : 2249-7455.

12. J.V.Anil Kumar, M. Amith, P. Lakshmi Usha Sri, K. Bewala, G. Kavya and SK.Abdul Munaf, "The Case Of Cross-Site Request Forgery And Machine Learning For Web Vulnerability", Detection.International Journal of Computer Engineering and Applications 16(9): pp.239-251.

13. J.V. Anil Kumar, Naru Kamalnath Reddy, Bollavaram Gopi, Derangula Akhil, Dareddy Indra Sena Reddy, Akkalaakhil , "Language-Based Phishing Threat Detection Using ML And Natural Language Processing", International Journal of Management, Technology And Engineering (IJMTE), Volume XV, Issue IV, April 2025, Page No : pp. 406-416, ISSN NO : 2249-7455, 2025.

14. J.V.Anil Kumar, Siddi Triveni, Yaragorla Sravya, Mancha Mancha. Venkata Aksh, Posani Lahari Priya, Grandhe Sirisha , "Tools For Database Migration", International Journal of Management, Technology And Engineering (IJMTE), Volume XV, Issue IV, April 2025, Page No : pp. 760-766, ISSN NO : 2249-7455, 2025.

15. SK Althaf HussainBasha, Shaik Yasmin Sulthana, "IOT Based Shutter Alarm Security System" Journal of Engineering Sciences (JES), Vol.11, Issue 7,July/2020, pp.1035-1045, ISSN No:0377-9254.

16. Sk Althaf Hussain Basha, A. Amrutavalli, Mekala Anjali Lavanya, Vanama Dhakshayayani Sriya, Grandhisila Jahnavi, Pari Chaitanya Lakshmi , "Cloud-Based Decision Support Systems For Business Data Intelligence", International Journal of Management, Technology And Engineering (IJMTE), Volume XV, Issue IV, APRIL 2025, Page No : 303-313, ISSN NO : 2249-7455, 2025.

17. Sk Althaf Hussain Basha, A. Amrutavalli, Mekala Anjali Lavanya, Vanama Dhakshayayani Sriya, Grandhisila Jahnavi, Pari Chaitanya Lakshmi , "Cloud-Based Decision Support Systems For Business Data Intelligence", International Journal of Management, Technology And Engineering (IJMTE), Volume XV, Issue IV, April 2025, Page No : 303-313, ISSN NO : 2249-7455, 2025.

18. Sk. Althaf Hussain Basha, G. Mahesh, Kokkera Krishnaveni, Gadde Koushika, Derangula Manasa, Yalla Pranavi, "Honeytrap-Enabled Cloud Security Framework For Preventing Network Breaches", International Journal of

Management, Technology And Engineering (IJMTE), Volume XV, Issue IV, April 2025, Page No : 453-463 , ISSN NO : 2249-7455, 2025.