Big data encrypting transmission framework for a multi-AGV system based on federated deep learning

Tongpo Zhang[1], Yuxin Wan[2], Migue Lopez-Benitez[3], Enggee Lim[4], Fei Ma[5] and Limin Yu[6]

1. School of advanced technology, Xi'an Jiaotong-Liverpool University, Suzhou, China, Tongpo.Zhang19@xjtlu.edu.cn

2. School of advanced technology, Xi'an Jiaotong-Liverpool University, Suzhou, China, Yuxin.Wan20@xjtlu.edu.cn

3. Electrical Engineering and Electronics, University of Liverpool, Liverpool, United Kingdom, M.Lopez-Benitez@liverpool.ac.uk

4. School of Science, Xi'an Jiaotong-Liverpool University, Suzhou, China, enggee.lim@xjtlu.edu.cn

5. School of advanced technology, Xi'an Jiaotong-Liverpool University, Suzhou, China, Fei.Ma@xjtlu.edu.cn

6. School of advanced technology, Xi'an Jiaotong-Liverpool University, Suzhou, China, Limin.Yu@xjtlu.edu.cn

***Abstract*—This project has developed a data transmission framework including Automated Guided Vehicles (AGVs), a cloud platform, edge computers and a master server. This framework is designed to provide a solution for protecting user privacy in the data transmission process of AGV in practical applications. The system divides the information collected by AGVs into messages and big data, where messages are transmitted through Message Queuing Telemetry Transport (MQTT) protocol and big data is transmitted through socket. Big data is collected by different sensors equipped in AGVs. Messages is processed by the AGV's main chip. Meanwhile, public and private key encryption based on RSA algorithm is performed in the transmission process to ensure the privacy and safety of information and users.**

Keywords-Big data;FL; multi-AGV system; RSA

I. INTRODUCTION

A multi-AGV system has been widely applied in industrial areas. The system contains many different sensors and chips. Big data and messages collected or processed by those sensors and chips should be transmitted to a central platform or a cloud platform. Furthermore, the safety of the communication environment should also be considered to establish a transmission framework.

To solve the above problems, we assume that there is a multi-AGV system that performs a series of tasks in a known environment. There are multiple task target points in the scene, a starting point and a task list. After accepting the task, each AGV starts from the starting point and moves to the task target point at a certain interval. After each AGV reaches the target point of the task, it completes the task and returns to the starting point. On the way, each AGV collects information in the environment through a variety of different sensors equipped by itself. The data collected by sensors equipped in AGV will be transformed to the central platform or the cloud platform depends on the specific situation. Messages processed by the AGV's main chip will also be transformed to the central platform or the cloud platform.

Based on the situations mentioned above, we establish a big data encrypting transmission framework for a multi-AGV system based on federated deep learning. This framework is created to improve a multi-AGV system's communication environment. Based on federated deep learning (FL) model, this framework simulates the scenario of big data transmission between the edge computer and the main server by realizing encrypted message transmission between multiple computers and different sensors [1].

For example, Computer A simulates an exchange node and computer B simulates a party node. Computer B can simulate multiple contracting nodes. The simulation process includes connecting multiple computers and different sensors through Alibaba cloud platform and realizing MQTT transmission. Specifically, computer A encrypts the message and uploads it to the cloud platform. After the cloud platform sends it, computer B receives and decrypts it. This realizes the simulation of short messages such as messages and parameters transmitted between AGVs and the main server. At the same time, the cloud platform receives and visualizes the model data to simulate the real-time supervision of AGVs. The RSA algorithm is used in the process. It is a public key encryption asymmetric algorithm and a standard for encrypting information transmitted through the

network. RSA encryption is powerful and reliable, it will generate a large number of chaotic codes [2].

The rest of the article is organized as follows: The related researches about federated deep learning model's application and RSA algorithm's applications are investigated in section 2. Section 3 describes the overviews of the structure of the designed framework. The detailed of testing results are in section 4. Lastly, this paper ends with conclusions and future research.

II. RELATED WORK

The federated deep learning model can be applied in different areas. Ahmed Imteaj and M. Hadi Amini proposes an FL model by monitoring client activities and leveraging available local computing resources, particularly for resource-constrained IoT devices (e.g., mobile robots), to accelerate the learning process and finally successfully avoid the inconsistent and inadequate resource clients from the training process [3]. Wei Liu's team propose a new federated learning design of Momentum Federated Learning (MFL) converges faster than FL for different machine learning models[4]. M. Gharibi and P. Rao raised a refining algorithm called RefinedFed, to eliminate corrupted, low accuracy, and noisy models that can negatively impact the centralized model by reducing its accuracy or cause other malicious activities [5]. Sherif B. Azmy's team applied a federated deep learning model to an UAV system to improve the system's efficiency [6]. Mohamed Amine Ferrag's team presents a comprehensive study with an experimental analysis of federated deep learning approaches for cyber security in the Internet of Things (IoT) applications [7].

RSA algorithm is raised by Ron Rivest, Adi Shamir and Leonard Adleman in 1977 [8]. It has a long history and is still regarded as the most popular public-key cryptography. Denning E Dorothy describes the attack with the RSA cryptosys tem and shows how it generalizes to other public-key systems [9].Ming-Der Shieh's team presents a new modular exponentiation architecture with a unified modular multiplication/square module and show how to reduce the number of input operands for the CSA tree by mathematical manipulation for RSA Cryptosystem [10]. Fu Mo's team presents an improved RSA cryptosystem based on chaotic synchronization which not only retains RSA's security features, but it can also significantly promote the security of traditional RSA cryptosystem [11]. Raza Imam's team summarizes and organizes recent literature results in a way that integrates and adds understanding to the work in the RSA field. They also emphasizes the classification of the existing literature to develop a perspective on the certain dominant RSA areas and domain and assess the research trends following the standard SLR methodology [12].

Our designed big data encrypting transmission framework for a multi-AGV system based on federated deep learning can be applied to edge devices, train the local data set with the model sent by the master server, encrypt the uploaded updated parameters, and send the updated model after the master server accepts the parameters.

III. APPROACH

The designed big data encrypting transmission framework for a multi-AGV system based on federated deep learning can be applied to edge devices, train the local data set with the model sent by the master server, encrypt the uploaded updated parameters, and send the updated model after the master server accepts the parameters.
The key factors of the frame work can be divided into three parts, the structure of the framework, message transmission and big data transmission.

A. The structure of Framework

The main content of the study, focus on building a system framework to provide a secure information flow scheme for AGVs in use. There are four components in the framework, which are the main server, the cloud platform, the edge computer, and a multi-AGV system, each AGV is equipped with different sensors. The structure of the framework can be seen in Figure 1: The Framework of the system.

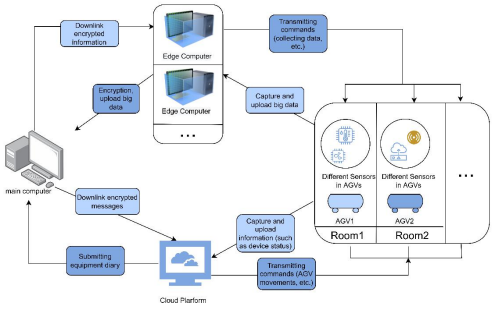

Figure 1. The Framework of the system

A multi-AGV system can be a system in a known environment or in many rooms. In the system, a main server is used to accept and decrypt big data messages and perform learning or model updates. It also encrypts and issues updated models. The cloud platform is used to receive and monitor the status information of AGVs (such as temperature, speed, etc.). It also issues commands and short messages. The edge computer is used as a computer to package and collect environmental information and data transmitted by AGVs (such as collect photo and data

set). The edge computer can also be used to train a local model with a local data set, encrypt and upload parameters. An AGV uploads the information collected by the sensors to the edge computer, and sends the material model information about the state received by the sensor to the cloud platform. An edge computer can correspond to one or more AGVs, and receive messages from different AGVs in the same region. Each AGV contains multiple sensors that send different types of data to the cloud platform and edge computers.

This project implements a simulation of the system. In the simulation process, there are four main targets: message transmission, message encryption and decryption, big data transfer, and big data encryption and decryption. The first goal is to implement multi-device connection to the cloud platform and send status information or other messages through MQTT. The cloud platform receives the information and then calls back the message. The second goal is to encrypt the message, using RSA algorithm to homomorphically encrypt the message, decrypt it after receiving the message. The third goal is to use socket long connections for big data transfers between devices [socket]. The fourth goal is to encrypt and decrypt big data with public and private keys to simulate encrypted file transfers between the primary server and the edge computer.

B. Message transmission

Messages processed by the AGV's main chip enables AGVs to be communicated with the cloud platform via the MQTT protocol. The message transmission process simulates message communication between the cloud platform and AGVs, and mainly plays a role in the following scenarios which can be seen in Figure 2: The flowchart of the message transmission.

1.An AGV sends a model message to the cloud platform, which is about the parameters collected by the A GV sensor (such as temperature, speed, etc.). The object model message is visualized in the cloud platform to achieve the supervision of AGVs by the cloud platform in order to issue instructions.

2.An AGV sends messages to the cloud platform. For example, the number of messages collected by the AGV, the number of uploaded pictures, and the reporting of faults in an AGV's emergencies.

3.The cloud platform issues instructions to AGVs to guide AGVs' acceleration, track change, information collection and other actions.

4.The cloud platform calls back the message to an AGV, such as the receiving status of the message, the difference between the number of collected messages and the number of accepted messages, etc.

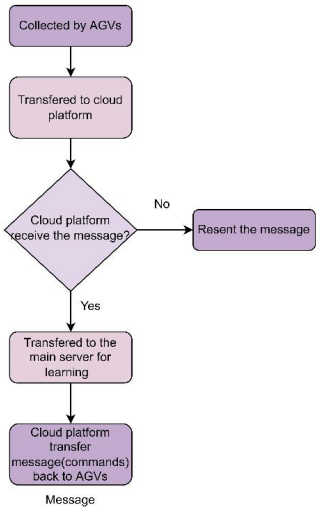

Figure 2. The flowchart of the message transmission

Since the environment in which the AGV is located may be resource-constrained, the AGV itself does not have strong processing power. Therefore, the MQTT protocol is used for transmission between AGVs and the cloud platform. MQTT is a machine-to-machine or Internet connection protocol over TCP/IP. MQTT is also a lightweight publish and subscription protocol, which is suitable for limited facilities situation, such as the low bandwidth and the high delay environment [13].

In order to make a connection between the cloud platform and the device and visualize the control of the device, the designed framework uses the Aliyun Cloud platform as the cloud platform in the framework to implement MQTT communication with the device, to achieve the control of the device and the transmission of the message.

Two scenarios are first used here to simulate three AGVs to connect to the cloud platform and implement a publish subscription by using three computers. The first option is to pass three computers through MQTT.fx connects to the cloud platform. MQTT.fx is an eclipse paho-based MQTT client written in the Java language that can be used to verify that a device can properly connect to IoT platforms and pass through a topic publish and subscribe messages. Use MQTT.fx to debug and test can help to observe and statistical equipment status.

The second option is to connect the cloud platform via Python for MQTT communication. We connect the device to Aliyun Cloud by importing the linkkit package required to connect to Aliyun cloud platform and perform publishing and subscription operations.

After the comparison we thought that connecting via python would perform better because of its ability to upload information with more flexible data types. However, not all AGVs contain such computing power since in the simulation we regard our laptop as an AGV. Therefore, using MQTT.fx in the case of poor environmental resources to connect the devices to aliyun platforms can be regarded as a better choice. When Python connects to Alibaba Cloud, the RSA library in Python can be used to encrypt the message upload and decrypt the information after receiving it. The RSA public key cryptography is a cryptographic system that uses different encryption keys and decryption keys, and it is not computationally feasible to derive the decryption key from a known encryption key. The RSA algorithm is described as follows:

(1) Arbitrarily select two different large prime numbers p and q to calculate

$$n = pq, \varphi(n) = (p-1)(q-1) \quad (1)$$

(2)Arbitrarily choose a large integer , e satisfied

$$\gcd(e, \varphi(n) = 1) \quad (2)$$

the integer e is used as a key.

(3) Determine the solution key e, satisfied

$$(de) mod \varphi(n) = 1 \quad (3)$$

(4) disclosure of integers n and e, secret preservation d;

(5) The plaintext m(m<n is an integer) is encrypted into ciphertext c, and the encryption algorithm is

$$c = E(m) = m^e mod n \quad (4)$$

(6) Decrypt ciphertext c to plaintext m, and the decryption algorithm is

$$m = D(e) = c^d mod n \quad (5)$$

RSA is the underlying algorithm in asymmetric cryptographic algorithms, and here only as an example of an encryption method [8]. The secrecy strength of the RSA algorithm increases as the length of its keys increases. The longer the key, the longer it takes to encrypt and decrypt, but it is suitable for many environments due to its high compatibility.

The application of encryption algorithms in the transmission of messages can strengthen the security of data. However, an AGV is limited by bandwidth delays in the process of transmitting messages, and an AGV has limited computing power. Therefore, Applying cryptographic algorithms may not be available for all AGV, but in environments or factories with high requirements for data security, the use of encryption algorithms is necessary.

C. Big data transmission

This part is for implementing big data encryption transmission socket between devices [14]. This process simulates data communication between the edge computer and the primary server and is primarily used in the following scenarios which can be seen in Figure 3 : The flowchart of Big data transmission.

1.The edge computer sorts the data set which collected by the sensor on the AGV and packages it. The edge computer also encrypts the packaged big data with the public key from the main server and uploads it to the main server.

2.The master server receives the big data and decrypts it with the local private key. Then the master server transmits the results obtained after local training or the updated model to the edge computer through the public key which sent by the edge computer.

3.After the edge computer receives data set uploaded by an AGV, it uses the primary server to train the data set locally. Then the edge computer encrypts the updated parameters or gradient data through the public key issued by the primary server. Finally, the edge computer uploads the data to the primary server.

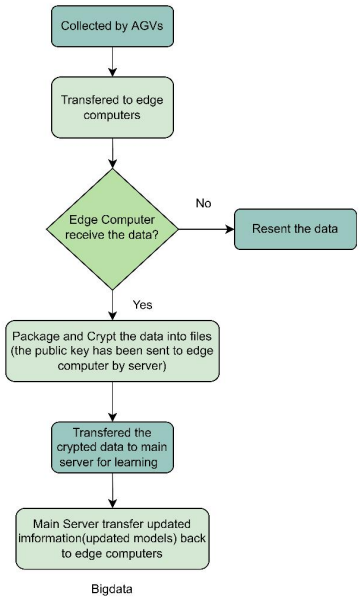

Figure 3. The flowchart of Big data transmission

The edge computer uploads parameters and gradient information through the model training data set issued by the main server, and the process of the main server receiving information and updating the model is based on the federated deep learning model. The learning process based on federated deep learning can better ensure data privacy, which can protect the original data collected by AGVs and calculated by edge computers. The data would not be

shared by the main computer and other edge computers. However, in the case of low data security requirements or the edge computer has computing performance limitations, encrypting the original data and uploading it can also ensure data security.

In the process of data transmission, we use socket long-distance transmission technology, which can realize real-time bidirectional communication of various communication mechanisms and has strong communication efficiency.

We use three computers to simulate the whole process of data transmission. Two computers serve as the edge calculator and one computer serves as the primary server. First, the master server downloads the public key used for encryption to both computers. The two computers package the dataset and encrypt it with the public key, upload the encrypted big data, and send different public key to the main server. After the main server receives the big data, it is decrypted with a private key, and after learning the data, the updated information and learning results are encrypted and uploaded to two different computers through the public key sent by the two computers. The two computers receive the message and decrypt it with their own private key, update the data or issue a new command to AGV.

We used the Cipher library in Python to complete the encryption decryption and file transfer of big data. The encryption algorithm used here is the RSA algorithm, which will use a different name for public.pem and private.pem files store public keys and private keys for each device. Because encryption has a limit on bytes, the file needs to be split. Similarly, when decrypting, the file needs to be split into paragraphs for decryption.

IV. EXPERIMENTS RESULTS

In order to verify the feasibility of the scheme and the performance of file transfer in practical applications, we simulated multiple devices with three computers (computer A, computer B, computer C), two of which (computer A, computer B) are closer and the other computer (computer C) is farther away from these two computers. The transferred data in the process are encrypted and decrypted basedon the public key and the private key. The public key of another device is used to encrypt and send the message to the device. The device that issued the public key accepts the information and decrypts it with the private key. The public and private keys are stored in .pem files. Public keys issued by different devices are distinguished by different file names. In this process, the public and private keys are randomly generated and written to a file. An example can be seen in Figure 4: An example of public key and Figure 5: An example of Public key.

Figure 4. An example of private key

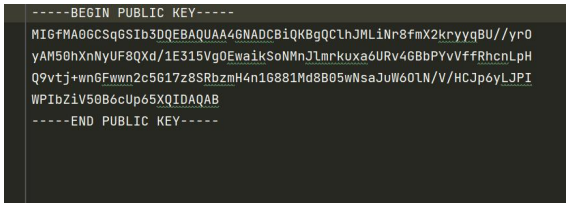

Figure 5. An example of public key

The tests are divided into the following categories:

1.In different Quality of Service situations, multiple devices simultaneously transmit messages to the Alibaba Cloud platform for messages and IoT messages, and analyze whether packet loss and queuing occur.

2.Test how multiple devices behave when sending encrypted big data to the same device.

The quality of service (Qos) in MQTT is mainly divided into three types, namely Qos0, Qos1 and Qos2, and its features and functions are as follows:

Since Aliyun cloud platform only supports Qos0 and Qos1 quality of service, and Qos2 is not suitable for this scenario, we will only test Qos1 and Qos2.

In Test 1, we tested the queuing, delay, and packet loss under different Qos in two ways. Log in to the cloud platform on one of the two closer computers (computer A) and receive MQTT on three computers fx (simulates a message sent by three AGVs).

The first is to have three computers send a message to the server every five seconds at the same time to see if it loses packets. We set several indicators to observe the test results. The first metric is the message loss rate, which is the difference between the number of messages sent and the number of received messages. In the second indicator is the time difference, i.e. the time difference between the pure time and the subscribe time.

We tested separately in different Qos cases.

The result is as follows:

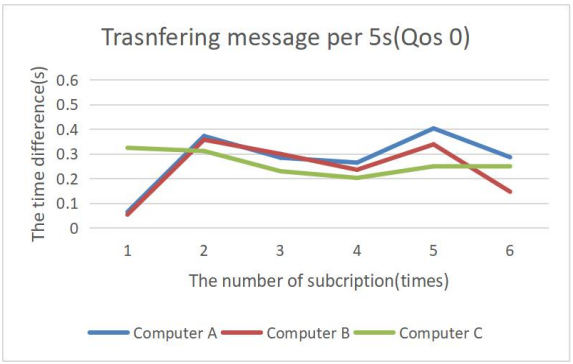

Figure 6. Transferring message per 5s (Qos 0)

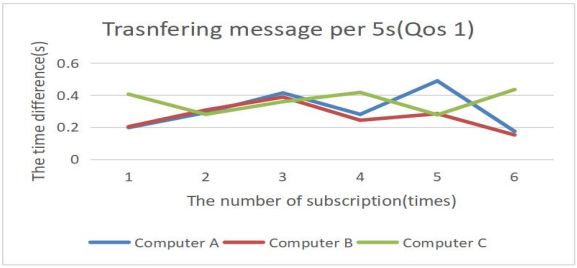

Figure 7. Results of Transferring message per 5s (Qos 1)

Figure 6 and 7 show the results of transferring message per 5s in Qos0 and Qos1. Six consecutive subscriptions were selected for separate analysis in the experiment. The ordinate is the time difference between publishment and subscription. The polyline represents the speed at which different computers subscribe messages under that Qos. The blue line shows the performance of computer A. The red line shows the performance of computer B. The green line shows the performance of computer C.

In Qos0 the average time difference is shorter than in Qos1. Additionally, the computer which are in the same area (Computer A and B) has the similar time difference while the green line (computer C) is different from others. With Qos0, the computer C, which is far from the cloud platform even has the shorter time difference in the number of subscription 2-5. This may because the network latency of computer C at that time was lower than that of computer A and B. In Qos1, computer C also did not have a higher time difference than computer A and B's. It means that the distance between the computer and the cloud platform is not strongly correlated to the time difference, which is the speed of information transmission. It would be effective for AGVs in different regions to adopt MQTT to communicate with the cloud platform.

TABLE I. THE STATISTICAL RESULT OF TEST 1

Name	Average time difference(s)	Packet loss (%)
Qos0	0.259056	0
Qos1	0.310833	0

As can be seen from the Table 1: The Statistical Result of Test 1, the average time difference in the case of Qos 0 is shorter, which is due to the higher the QoS level, the more packets are switched, increasing the end-to-end latency. The distance between computers has no obvious impact on indicators such as time difference. At the same time, due to the short and low frequency of transmission messages, the error of the number of messages is close to 0 under the two quality of service in the test, which shows a better transmission ability. Secondly, we use three computers to send a continuous large amount of information to a computer in a short period of time to observe its packet loss and queue. The above packet loss rate and time difference need to be used as an indicator here.

Tests were conducted separately in different Qos cases and the results were as follows:

Figure 8. Results of Transferring message in a short time (Qos 0)

Figure 9. Results of Transferring message in a short time (Qos 1)

Figure 8 and 9 show one of the real result of transferring messages in a short time in different Qos condition. The device "phone" responds to computer A, "A_MQTTfx" respond to computer B and "Mac_OS" responds to computer. The figure shows the data saved in Alibaba Cloud logs. In the case of Q os0, there is a short time difference, and in the case of multiple devices sending a large amount of information at the same time, it will be possible to receive messages from the same device continuously.

However, this does not occur with Qos 1 with Quality of Service, and messages sent by different devices will be received in a queued manner. At the same time, since Qos1 will continue to send the same message when it does not receive a postback message, message redundancy will occur in the results.

TABLE II. THE STATISTICAL RESULT OF TEST 2

Name	Average time difference(s)	Packet loss (%)
Qos0	0.19667	0
Qos1	0.2134	0

As can be seen from the Table 2: The Statistical Result of Test 2 and icons, under the test of both quality of service, the error in the number of messages is 0, presumably because of the small number of devices and low network latency [15]. In Test Two, we test the transmission and acceptance of big data after encryption. where computer A as main computer, computer B and computer C as edge computer send files to computer A. Different devices need to be connected to the same LAN in socket transmissions, so there is no distance difference between computers in this test. We have Computer B and Computer C continuously send large encrypted files to Computer A, which decrypts them after accepting them. Observe how computer A receives files and decrypts them.

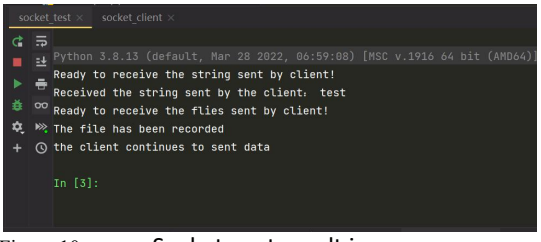

Figure 10. Socket sent result in server

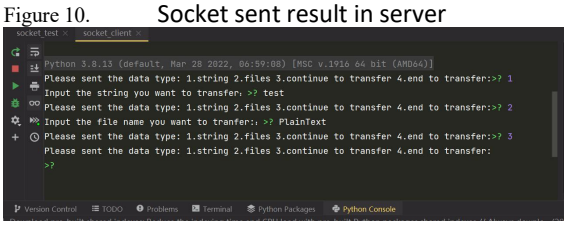

Figure 11. Socket sent result in client

Figure 10 and 11 show the process of transferring files over sockets on server and client. Due to the time required for the transfer of big data, a queue condition needs to be set up in the program. In this test we set the program that the file needs to be sent after the previous file is sent before it can continue. In some scenarios that require system preference, it is equally necessary to set the file priority of the device.

After the above series of tests, we successfully completed the transmission of messages under different Qos and the transmission of big files through sockets. By using MQTT upload status, issue instructions, and socket transmission of encrypted files, the server receives encrypted files and decrypts justified the rationality of the framework.

V. CONCLUSION AND FUTURE WORK

In summary, we establish a whole Big data encrypting transmission framework for a multi-AGV system based on federated deep learning. The system takes the message and big files apart, transmits the message through the cloud platform and transmits the big file through socket by the edge computers. Adding encryption and decryption of big data to the transmission can ensure the real-time nature of the message and the privacy of the big data at the same time.

The testing results show the designed framework It can realize the message and object model message transmission between AGV and the cloud platform through MQTT to achieve real-time monitoring. It can also enable communication between devices in different network environments and cloud platforms. At the same time, it can realize the transfer of large files and the encryption and decryption of public and private keys from multiple devices to the same device The framework can run smoothly in the simulation environment and has good performance. This framework can be used in diversity environment.

In the future, we plan to improve and apply the design framework to a real multi-AGV system to help the system to own a better communication environment.

ACKNOWLEDGMENT

This research was supported by the Research Enhancement Fund of XJTLU (REF-19-01-04).

REFERENCES

[1] K. M. Ahmed, A. Imteaj and M. H. Amini, "Federated Deep Learning for Heterogeneous Edge Computing," 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), 2021, pp. 1146-1152, doi: 10.1109/ICMLA52953.2021.00187.

[2] D. Boneh and G. Durfee, "Cryptanalysis of RSA with private key d less than N/sup 0.292/," in IEEE Transactions on Information Theory, vol. 46, no. 4, pp. 1339-1349, July 2000, doi: 10.1109/18.850673.

[3] A Imteaj ,M H Amini . FedAR: Activity and Resource-Aware Federated Learning Model for Distributed Mobile Robots[J]. 2021.

[4] W Liu ,L Chen ,Y Chen, et al. Accelerating Federated Learning via Momentum Gradient Descent[J]. 2019.

[5] M. Gharibi and P. Rao, "RefinedFed: A Refining Algorithm for Federated Learning," 2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), 2020, pp. 1-5, doi: 10.1109/AIPR50011.2020.9425094.

[6] S. B. Azmy, A. Abutuleb, S. Sorour, N. Zorba and H. S. Hassanein, "Optimal Transport for UAV D2D Distributed Learning: Example using Federated Learning," ICC 2021 - IEEE International Conference on Communications, 2021, pp. 1-6, doi: 10.1109/ICC42927.2021.9500304.

[7] M. A. Ferrag, O. Friha, L. Maglaras, H. Janicke and L. Shu, "Federated Deep Learning for Cyber Security in the Internet of Things: Concepts, Applications, and Experimental Analysis," in IEEE Access, vol. 9, pp. 138509-138542, 2021, doi: 10.1109/ACCESS.2021.3118642.

[8] Z. Xin and T. Xiaofei "Research and implementation of RSA algorithm for encryption and decryption," Proceedings of 2011 6th International Forum on Strategic Technology, 2011, pp. 1118-1121, doi: 10.1109/IFOST.2011.6021216.

[9] D., Dorothy E . Digital Signatures with RSA and Other Public-Key Cryptosystems[J]. Communications of the Acm, 1984, 27(4):388-392.

[10] M D Shieh , J H Chen ,H H Wu , et al. A New Modular Exponentiation Architecture for Efficient Design of RSA Cryptosystem[J]. IEEE Transactions on Very Large Scale Integration Systems, 2008, 16(9):1151-1161.

[11] F. Mo, Y. -C. Hsu, H. -H. Chang, S. -C. Pan, J. -J. Yan and T. -L. Liao, "Design of an Improved RSA Cryptosystem Based on Synchronization of Discrete Chaotic Systems," 2016 International Conference on Information System and Artificial Intelligence (ISAI), 2016, pp. 9-13, doi: 10.1109/ISAI.2016.0012.

[12] R. Imam, Q. M. Areeb, A. Alturki and F. Anwer, "Systematic and Critical Review of RSA Based Public Key Cryptographic Schemes: Past and Present Status," in IEEE Access, vol. 9, pp. 155949-155976, 2021, doi: 10.1109/ACCESS.2021.3129224.

[13] B. Mishra and A. Kertesz, "The Use of MQTT in M2M and IoT Systems: A Survey," in IEEE Access, vol. 8, pp. 201071-201086, 2020, doi: 10.1109/ACCESS.2020.3035849.

[14] Y. Yubing and L. Zhanping, "Research of fast and safe large files transmission based on Socket," 2012 7th International Conference on Computer Science & Education (ICCSE), 2012, pp. 483-485, doi: 10.1109/ICCSE.2012.6295119.

[15] S. Lee, H. Kim, D. -k. Hong and H. Ju, "Correlation analysis of MQTT loss and delay according to QoS level," The International Conference on Information Networking 2013 (ICOIN), 2013, pp. 714-717, doi: 10.1109/ICOIN.2013.649671