

**COMP90015**

**Distributed System**

**Project 2**

**Group ZWPL**

**Group Members**

Jialiang Zhu jialiangz@student.unimelb.edu.au

Yanzi Wang yanziw@student.unimelb.edu.au

YunqiangPu yunqiangp@student.unimelb.edu.au

MiaozeL miaozel@student.unimelb.edu.au

**Encryption and file permission scheme**

There are many potential ways to make sure the security of the file permission scheme in BitBox. Generally, the potential threats can be concluded as “Leakage”, “Tempering” and “Vandalism”. That is to say, all the security in the BitBox can be divided into two parts, one is the file permission scheme before the transmission, and another is the security during the transmission.

As for the solution, it can be divided into two parts. The first way is to verify the operating authorization before building effective transmission which can be concluded as ownership and access control. the second way is to send all the information to the public channel without verification, and only the target receiver with corresponding key can decrypt the message, this method can be concluded as encryption methods. Both methods have advantage and weakness. Either of them cannot make sure the BitBox run securely and confidentially. Here is the detailed discussion of two methods as following:

1. Ownership and access control

This method is designed to address the potential security problem in peer-to-peer of “leak” that only identity user with the authorization can access and update the specify file and directory. The progress can be concluded as three steps. Firstly, the file/directory is created by the user by FILE\_CREATE\_REQUEST/DIRECTORY\_CREATE\_REQUEST command. After that the owner identity is given to the creator together with the reading and writing privilege of this file/directory. Secondly, the access/update permission of this file/directory can be given to other users via manually operation to the access-control list by the owner. Lastly, when every client send request to access or update the file/directory it will check the client’s identity and return an ACCESS\_UPDATE\_RESPONSE status with true/false status to represent the result of the verification of this operation authorization. In this way, it makes sure the only the specified client can have operation authorization to specified file/directory.

1. The encryption methods can be concluded as three methods that public key, private key and the Advanced Encryption Standard. The principle of the public key and private key is that, after the ssh-keygen method being created they are given to the potential sender and receiver respectively which are 1024 to 2048 bits(small and easy to store). Then all the senders can send the plaintext decrypted by the public key. Only the receiver with the corresponding private key can decrypt the [ciphertext](https://fanyi.sogou.com/?keyword=ciphertext&fr=websearch_submit&from=en&to=zh-CHS) to the plaintext. By this method, the confidential communication between sender and receiver can be guaranteed. As for the Advanced Encryption Standard (AES), after it has been created by the server, it is encrypted by the public key to the corresponding client and after the client receiving and decrypting. A security communication of AES (128) established security. After the three kinds of commands exchange successfully, the connection was lost and the AES will restart in a new connection.

**Limitations**

In order to protect the files and messages shared in communication. Various kinds of encryption methods exist. What we will talk about here is the public/private keys related encryption. Although the public/private key can play a role in security protection, there are also some limitations in some condition. In the following contents, we will give some examples, and provide the justifications to briefly explain the reasons.

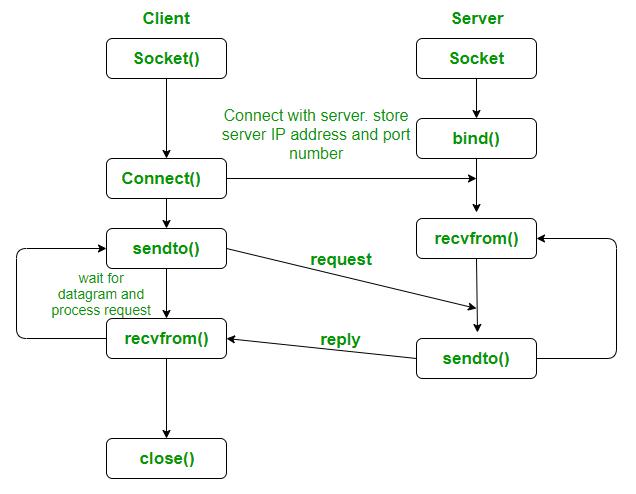
1. As the theory of how the public/private key encryption works, the process of encryption and decryption need to use two keys. The public key for encryption, and the private key for decryption. With comparing it to the secret key encryption which just needs one key during the process, the public/private key encryption seems to have a slower speed. The process of public/private key method will cost more time and resources of computing to encrypt and decrypt.
2. Another limitation is about the authentication issues. As there are no methods in public key encryption to verify the identity of the user. It is possible for the transmitted message to be intercepted by someone else without the sender’s knowledge. And the receiver would not know whether the message received is the authentic one and who sent the file out.
3. There is also another potential risk related to the private key transfer should be taken into consideration. As the one who needs to use the private key to do decryption for received files, the access to the key itself need to be gained before the operation. This requires the transmission of the key to the one who needs access. However, it is not secure as others may intercept the transmission and steal the access, even pretend to be the one who was supposed to receive the access, tamper the data.
4. The fourth limitation is about the number of private keys. If there is only one private key used in the encryption system, others can decrypt all the data after you share the private key with them, even though you just want to share only part of the data. In this case, if you want to separate the information into several parts, multiple private keys are needed to be generated for the information groups. This may also lead to an extra cost of time efforts as it increases the number of operations for encryption and decryption.

**Protocols**

We should choose the protocol according to the situation. As for UDP, if the network quality is very unsatisfactory, packet loss will be a great problem.

By using UDP, data is sent regardless of the status of the other peer. Since the transmission data does not establish a connection, there is no need to maintain the connection status, including the transmission and reception status, so that one server can transmit the same message to multiple clients at the same time.

Messages should be added the UDP header, when they transmitted from application layer to transport layer. In addition, the IP header should also be added, when messages are transmitted to IP layer. In the whole process, the messages handed over by the application layer are neither merged nor split, so we can get the boundaries of these messages. Furthermore, when the application layer delivers the packet, the packet will be sent as it is, that is, one packet is sent at a time. As for the recipient, the header is removed at IP layer, and the rest of the message is sent to the next layer. The following chart is a demonstration of the interaction diagram:



UDP packet contains 8-byte UDP header and the data we want to send. The header contains source port, destination port, packet length and checksum, each part is 2-byte. IP packet contains 20-byte IP header and UDP packet.

**The Aspect of our UDP implementation and the comparison with TCP**

The packet loss in UDP is a critical issue when designing the file transferring system because of the lack of acknowledge. When the packet loss and the file transfer fail, it is unable for the system to detect the fault and start a re-transmit. Therefore, addressing the packet loss issue is one of the most significant measure to ensure the quality of the data transmitting and the overall system. In order to achieve the re-transmission function, here are the steps we implemented in our BitBox Peer UDP. We first created a list to store all the request sent from the peers. A thread which using the Boolean parameter is implemented to listen to the list. When a response is received from the peer, the store information in the list is deleted. Otherwise，if the list’s length is long than zero, the system will sent the same request again after waiting for a period of time. Using the time-out methods for this list, we are able to solve the packet loss issue for the UDP protocol for the BitBox Peer.

Compare with the TCP protocol, the UDP protocol is able to provide the same functionality with the implementation of the time-out list. However, certain disadvantage exists for the UDP protocol. One disadvantage is that the UDP seems to have less security measure. Because UDP transmission does not require the connection to be established, it is hard to ‘trust’ some peer’s IP address over the others. Also, since no states is kept for UDP protocol, unauthorized packets could be sent by the attacker. In the meanwhile, there lacks flow control for UDP. Nevertheless, it does provide certain advantages as well. UDP is a faster method compare with TCP since it does not need to connection establishment time. Also, it suits the system when the clients number scales better then TCP if the queries type from the clients are fixed and simple.