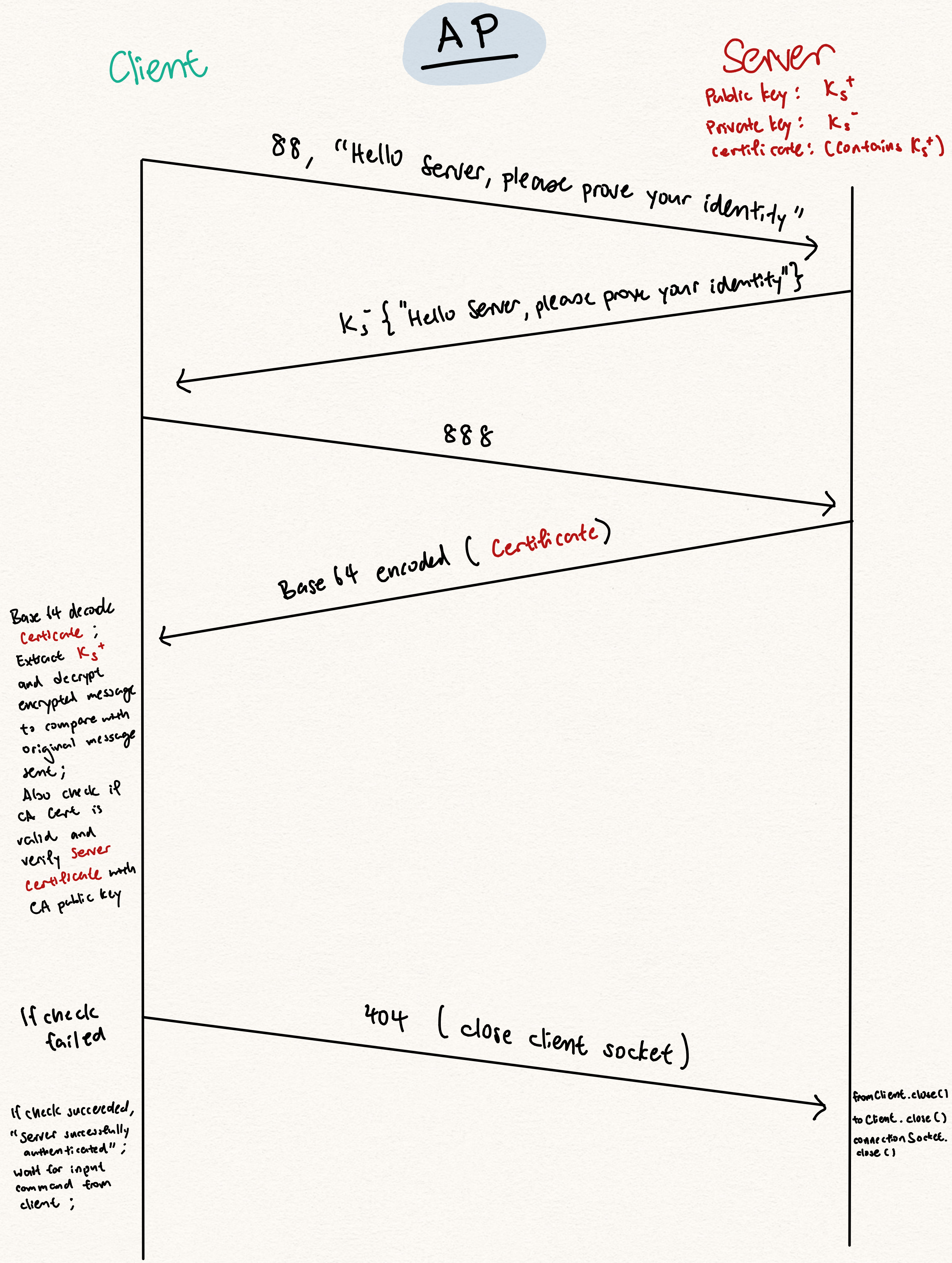
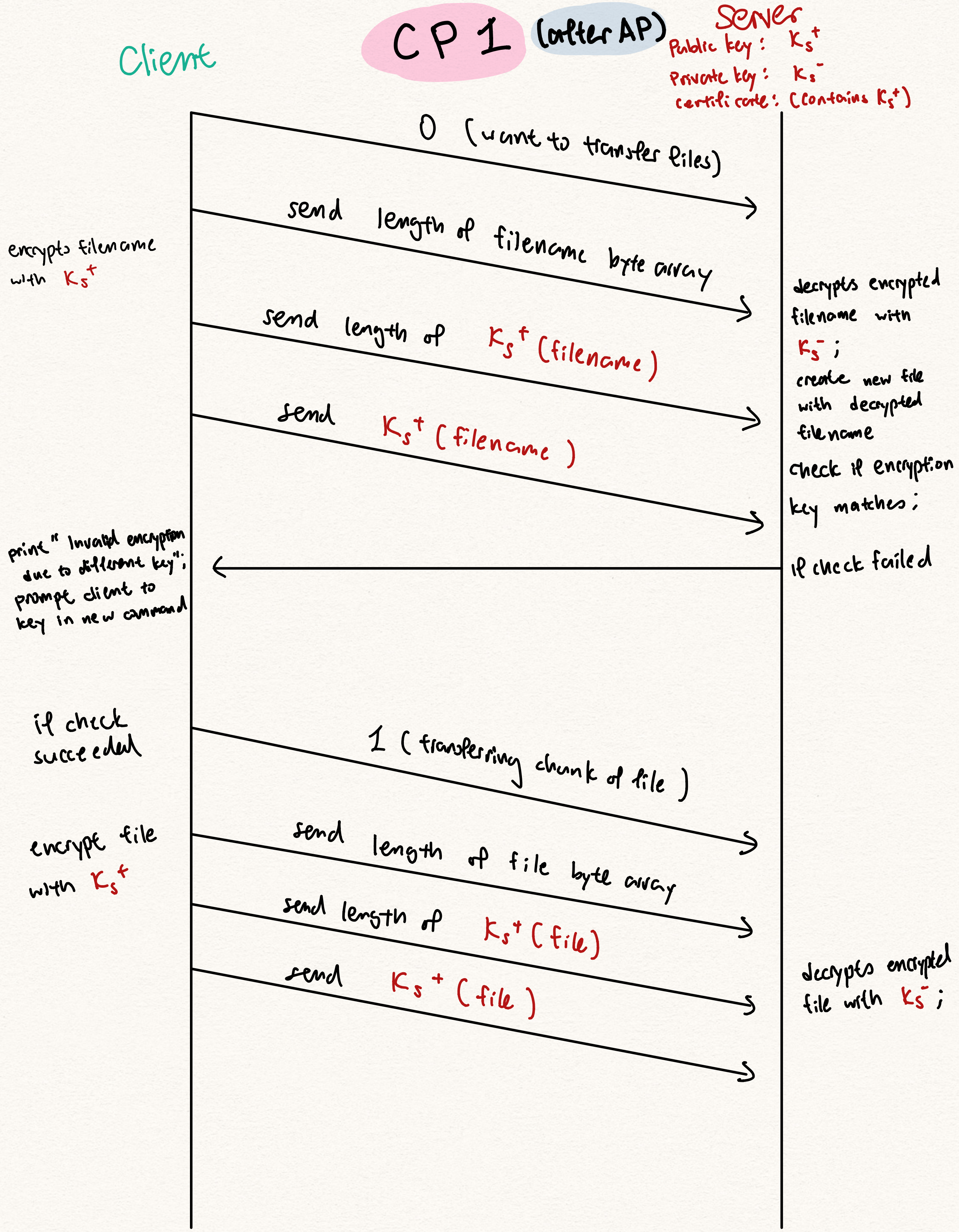
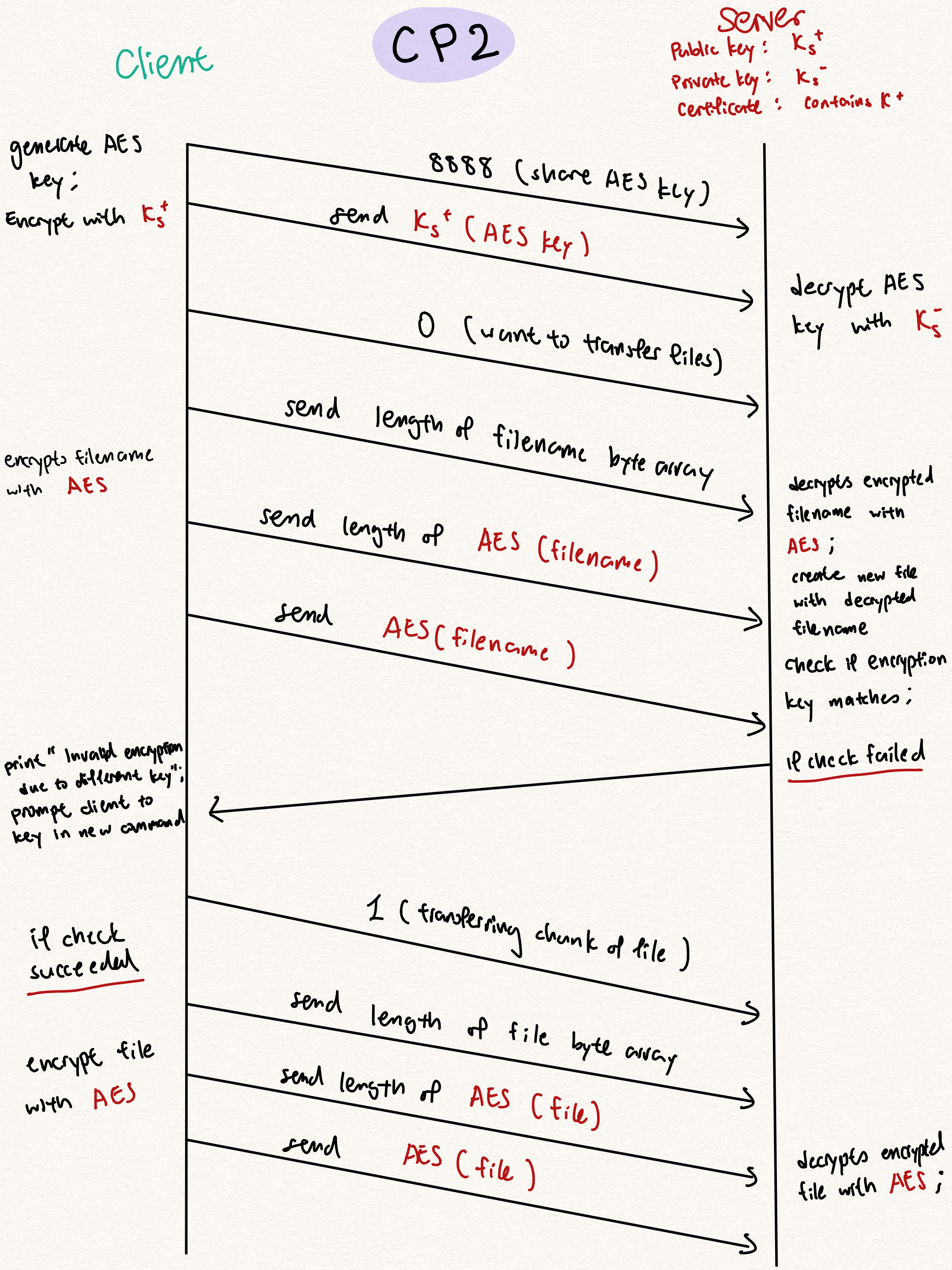

AP
Client
Server
Public key : K_s^+
Private key : K_s^-
certificate : (contains K_s^+)
88, "Hello server, please prove your identity"
K_s^- { "Hello Server, please prove your identity" }
888
Base 64 encoded (Certificate)
Base 64 decode Certicate ; Extract K_s^+ and decrypt encrypted message to compare with original message sent ; Also check if CA cert is valid and verify Server certificate with CA public key
If check failed
404 (close client socket)
fromClient.close()
toClient.close()
connectionSocket.close()
If check succeeded, "Server successfully authenticated" ; wait for input command from client ;

CP 1 (after AP)


Client
Server
Public key : K_s^+
Private key : K_s^-
certificate : (contains K_s^+)
0 (want to transfer files)
encrypts filename with K_s^+
send length of filename byte array
send length of K_s^+(filename)
send K_s^+(filename)
decrypts encrypted filename with K_s^-;
create new file with decrypted filename
check if encryption key matches;
if check failed
print "Invalid encryption due to different key"; prompt client to key in new command
if check succeeded
1 (transferring chunk of file)
encrypt file with K_s^+
send length of file byte array
send length of K_s^+(file)
send K_s^+(file)
decrypts encrypted file with K_s^-;


CP2
Client
Server
Public key : K_s^+
Private key : K_s^-
Certificate : contains K^+
generate AES key;
Encrypt with K_s^+
8888 (share AES key)
send K_s^+ (AES key)
decrypt AES key with K_s^-
0 (want to transfer files)
send length of filename byte array
encrypts filename with AES
send length of AES (filename)
send AES(filename)
decrypts encrypted filename with AES;
create new file with decrypted filename
check if encryption key matches;
if check failed
print "Invalid encryption due to different key";
prompt client to key in new command
1 (transferring chunk of file)
if check succeeded
send length of file byte array
encrypt file with AES
send length of AES (file)
send AES (file)
decrypts encrypted file with AES;