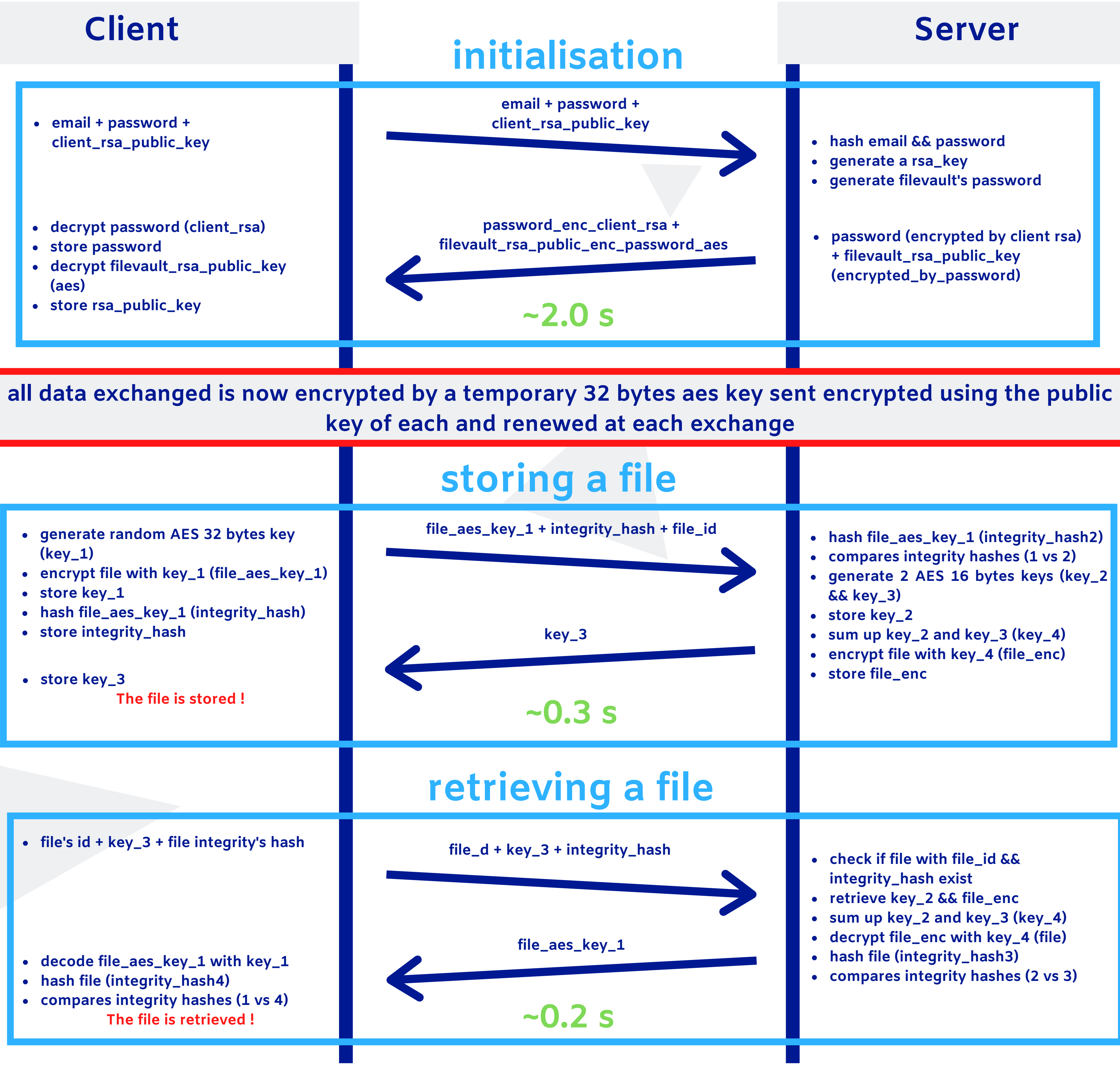
FileVault

Safely store your files, none of the part got the full key

Decoding each file would take 2,3 x 10^68 years at ... one billion tries per second !

230 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 years



Client | Server

initialisation

Client:

- email + password + client_rsa_public_key

email + password + client_rsa_public_key →

Server:

- hash email && password
- generate a rsa_key
- generate filevault's password

← password_enc_client_rsa + filevault_rsa_public_enc_password_aes

Server:

- password (encrypted by client rsa) + filevault_rsa_public_key (encrypted_by_password)

Client:

- decrypt password (client_rsa)
- store password
- decrypt filevault_rsa_public_key (aes)
- store rsa_public_key

~2.0 s

all data exchanged is now encrypted by a temporary 32 bytes aes key sent encrypted using the public key of each and renewed at each exchange

storing a file

Client:

- generate random AES 32 bytes key (key_1)
- encrypt file with key_1 (file_aes_key_1)
- store key_1
- hash file_aes_key_1 (integrity_hash)
- store integrity_hash

file_aes_key_1 + integrity_hash + file_id →

Server:

- hash file_aes_key_1 (integrity_hash2)
- compares integrity hashes (1 vs 2)
- generate 2 AES 16 bytes keys (key_2 && key_3)
- store key_2
- sum up key_2 and key_3 (key_4)
- encrypt file with key_4 (file_enc)
- store file_enc

← key_3

Client:

- store key_3

The file is stored !

~0.3 s

retrieving a file

Client:

- file's id + key_3 + file integrity's hash

file_d + key_3 + integrity_hash →

Server:

- check if file with file_id && integrity_hash exist
- retrieve key_2 && file_enc
- sum up key_2 and key_3 (key_4)
- decrypt file_enc with key_4 (file)
- hash file (integrity_hash3)
- compares integrity hashes (2 vs 3)

← file_aes_key_1

Client:

- decode file_aes_key_1 with key_1
- hash file (integrity_hash4)
- compares integrity hashes (1 vs 4)

The file is retrieved !

~0.2 s

Calculation: each file is encoded using 3 AES keys (32, 16 and 16 bytes)

2^256 x 2^128 x 2^128 = 2.318 x 10^77 possibilities

client got 3/4 of the full key, server got 1/4 of the full key and the file

Courtel Eliot | Sept 2020