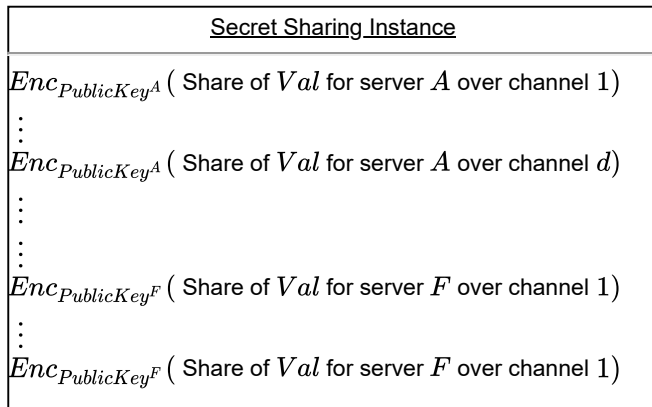


# Anonymous Chat Architecture

Sending a message  $M$  from client  $X$  to client  $Y$



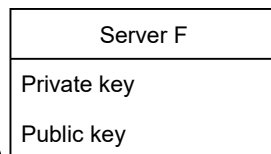
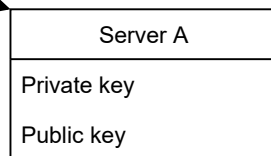
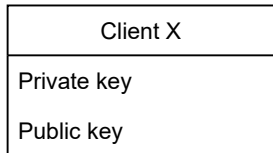
The instance is replicated(via Raft) among all nodes, allowing recovery in case of client/server omissions - if the client sent the instance to at least one valid server, recovery is guaranteed.

Despite replication, each server can only decipher shares intended for him due to use of PKI



Servers running anonymous secret sharing algorithm(of last exercise)

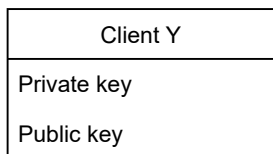
1. Secret Share Instance  
 $Val = Enc_{PublicKey^Y}(M; X)$



2. Replication of shares  
 During this phase, each server decrypts  $d$  different shares of a given client.

3. Recovery of  $Val$  (once a majority of clients sent their shares and they were replicated)

4. Broadcasting  $Val$  or Collision



5. Every client tries to decipher  $Val$ , only 1 will succeed

5. Client  $Y$  deciphers broadcast message  $M$ ,  $Sender = Dec_{PrivateKey^Y}(Val)$

He can now respond to  $X$ , repeating the procedure

Note the above procedure only talked about a single  $Val$ . In practice, servers wait for a majority of clients to send values, and there might be collisions, so a client might have to try multiple times before his message is sent successfully. In addition to  $Val$ , client can concatenate some randomly generated message identifier(not encrypted) to tell it was broadcast successfully. (if not, the sender is surely faulty - while technically we aren't required to handle this case, from a UX standpoint it would be nice to tell him that)