PQC Multi-Party Key Exchange (MP-KEX)

Authors: Sanish Gurung & Valtteri Niemi, University of Helsinki

Requirements for MP-KEX

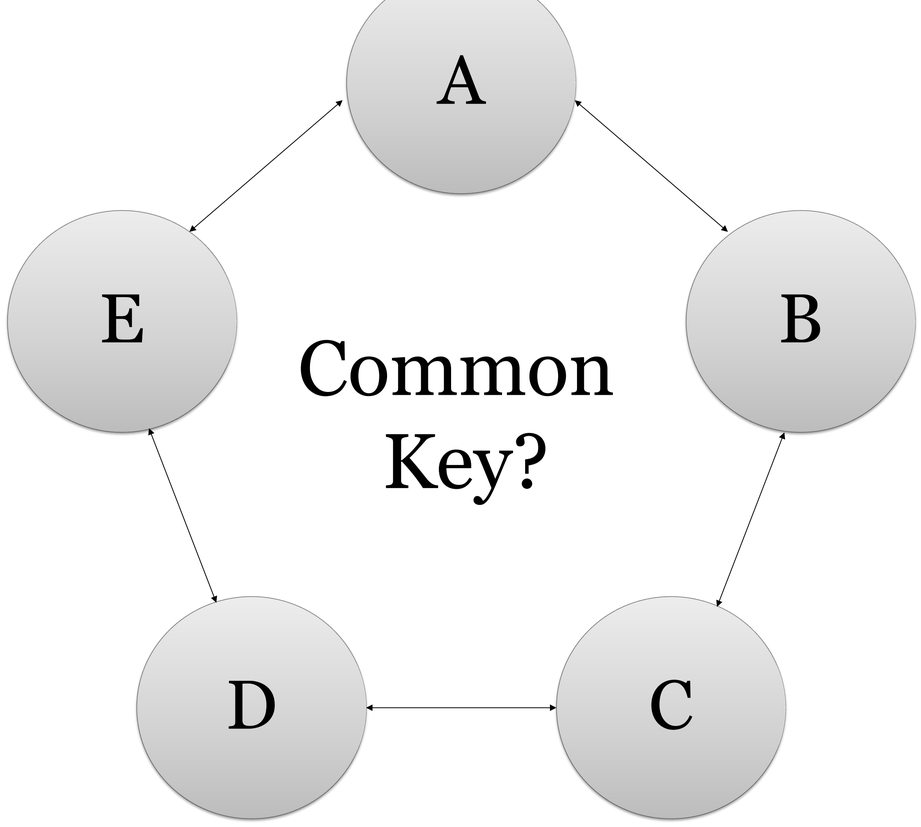
- **Find shared key between parties**
- No trusted server
- No leader, everybody does the same

Assumptions on parties

- Know signature verification key of every other participants
- Arranged to a cycle by lexicographical order of public keys
- Knows the algorithms of the protocol, including hashing, DSA, KEM, ECDH

Use Cases

- Tactical Military Networks
- E2EE Video Conferencing
- Blockchain Validator Committees
- Secure Federated Learning



Burmester-Desmedt with DLP (Elliptic Curve)

1. Party i chooses secret number r_i and multiplies with generator G to obtain public key $Z_i = r_i . G$. Then, Z_i is sent to neighbors.
2. Party i computes intermediate keys $K_{i-1,i} = r_i . Z_{i-1}$ and $K_{i,i+1} = r_i . Z_{i+1}$. Then, computes and broadcasts the *bridge* $X_i = K_{i,i+1} - K_{i-1,i}$.
3. Once all the bridges are obtained, party i computes the final shared secret key $K_{final} = \left(n \cdot K_{i-1,i}\right) + \sum_{j=0}^{n-2}(n - 1 - j) \cdot X_{i+j} = \sum_{j=1}^{n} K_{j,j+1}$.

Our Solution with ML-KEM

1. Party i acts as encapsulator and party $i+1$ acts as decapsulator, to share intermediate key $K_{i,i+1}$. First, party $i+1$ has to send its public key to party i.
2. Now, party i has intermediate shared secret keys with their neighbors, they compute and broadcast the bridge $X_i = K_{i,i+1} \oplus K_{i-1,i}$.
3. Once party i has received all bridges, they start retrieving all intermediate keys by computing $K_{j,j+1} = X_j \oplus K_{j-1,j}$ in the order $j = i+1, \ldots, i+n-2$. After retrieving all the intermediate shared secret key, everyone computes final shared secret key $K_{final} = H\left(K_{1,2} \parallel K_{2,3} \parallel \cdots \parallel K_{n,1}\right)$.

Performance Comparison