Impersonation Resistance in Module-LWE PAKE

Artifact Appendix

May 6, 2025

Theorem I: Impersonation Resistance

Statement.
If the PAKE protocol includes password-based authentication tied to the underlying LWE instance, then an adversary cannot impersonate a legitimate party without knowing the correct password.

Proof Sketch.
Impersonation attacks involve a malicious adversary $\mathcal{A}$ attempting to convince an honest party that it is another legitimate user. We analyze this under the assumption that passwords are used to derive or mask the LWE secrets:

1. The protocol embeds the password into the key derivation function or noise vector used in computing $B = A \cdot s + e$.
2. The honest party uses the same password-derived material to derive the shared key from the peer's public message.
3. If $\mathcal{A}$ does not know the correct password, any public message it sends will not allow the honest party to derive a matching session key.
4. As reconciliation produces inconsistent bitstrings, the hash of those bits results in a key mismatch.
5. Under Module-LWE, the adversary cannot guess or derive a valid s or e that results in successful authentication.

Conclusion.
Without access to the correct password, an adversary cannot compute or simulate a valid public value that leads to matching keys. Thus, the protocol resists impersonation attacks under standard assumptions.

$\square$

Theorem II: Key Confirmation

Statement.
If the PAKE protocol includes explicit key confirmation, then each party is assured that the other party has derived the same session key, provided the underlying Module-LWE problem is hard.

Proof Sketch.
Key confirmation is a mechanism that allows both communicating parties to confirm that they have derived the same session key. We analyze the correctness and security of this mechanism in the context of a Module-LWE-based PAKE protocol:

1. After computing the session key $K = \mathsf{H}(\mathsf{Rec}(u))$, each party sends a confirmation message, such as $\mathsf{MAC}_K("OK")$.

2. The recipient verifies the received MAC using their own derived key. If it succeeds, the recipient is assured the sender derived the same K.

3. Since K is derived from the LWE samples and ephemeral secrets, the adversary cannot forge MAC_K without knowing K.

4. If the adversary sends a bogus confirmation, it will not validate unless their derived key matches.

5. The only way an adversary can derive a valid K is by knowing the correct ephemeral secrets or solving Module-LWE.

Conclusion.
Under the hardness of Module-LWE and assuming the use of a secure MAC, the explicit key confirmation step guarantees mutual agreement on the session key.

$\square$

Theorem III: Known-Key Security

Statement.
If the Module-LWE PAKE protocol uses independent randomness and fresh ephemeral values for each session, then knowledge of one session key does not compromise the security of other session keys.

Proof Sketch.
Known-key attacks assume that the adversary has obtained session keys from previous protocol runs and attempts to use this information to derive other session keys or secrets.

1. Each session in the PAKE protocol independently samples $\mathbf{s}_A, \mathbf{s}_B$ and generates fresh noise vectors.

2. Public values B_A, B_B are thus independently computed from independent ephemeral secrets for each session.

3. Knowing a derived key $K_i = \mathsf{H}(\mathsf{Rec}(\langle \mathbf{s}_A^i, A \cdot \mathbf{s}_B^i + \mathbf{e}_B^i \rangle))$ gives no useful information about the ephemeral secrets or LWE values of another session.

4. Due to the preimage resistance of H and Rec, and the entropy in ephemeral secrets, K_j for $j \neq i$ remains pseudorandom even with full knowledge of K_i.

5. Recovering $\mathbf{s}_A^j$ or $\mathbf{s}_B^j$ from K_i or LWE samples would require breaking the LWE assumption.

Conclusion.
The PAKE protocol resists known-key attacks because each session key is generated independently, and recovering any session key does not weaken the security of other sessions.