Alan Palayil

A20447935

Homework #5

ECE 543

Due Date: 4/25/2023

12.1) Each login session has a unique Lamport Hash value, making it virtually impossible to predict future hash values from previous ones.

12.2) The transmitted value x, based on n and Alice's password can be calculated by an attacker who can then guess Alice's password and create candidate x values.

12.3) Bob compares the hash of Alice's transmitted quantity to the database value. If Alice doesn't send the same value as previously used in $hash^N$ calculation, the comparison fails.

12.6) In PDM, if either party selects x with $2^x < p$, an attacker can guess the session key by computing K with guessed W. To avoid this, both parties must ensure the received value isn't a power of 2.

12.7) Instead of storing W, use a public key system with private and public keys derived from W. Perform EKE/SPEKE based on public(W) and have the client sign hash(K) with private(W), which the server verifies. Encrypt the signature with K, if necessary.

12.8) Alice calculates $2^{ab} \bmod p$ by raising $(2^b \bmod p)^a$. To compute $2^{bW} \bmod p$, she raises $(2^b \bmod p)^W$, requiring W knowledge instead of $2^W \bmod p$.

12.9) Only Bob, knowing the correct modulus p, can compute b and $2^b \bmod p$. An imposter can't compute the first hash correctly, and Alice would detect it. An attacker with Bob's database access still can't compute Alice's response hash without W.

12.11) Alice computes K as: $K = ((g^b + g^w) - g^w)^{(a + aw)} \bmod p$.
Bob computes K as: $K = (g^a)^b (g^w)^{ab} \bmod p$.

12.14) Bob doesn't need to commit to a specific password in message 2. He can test passwords by decrypting Alice's encrypted exponential and raising it to the power b. To secure the protocol without Bob encrypting his Diffie-Hellman number, Bob should prove his knowledge of K before Alice does, either through a challenge-response or by sending a hash of K in message 2.

20.1) A local exploder creates a recipient roaster by checking and adding unique recipient from the message, including distribution lists, and sends the message to each recipient. A remote exploder sends the message to each recipient and relies on the remote exploder to eliminate duplicates. Efficiency is improved by including a message digest, which helps remote exploders remember and discard repeated messages.

20.2) Computing the message digest (MD) only once saves time and effort. This also applies to keyed MDs if a new per-message secret S is chosen, computed, and then encrypted with each recipient's secret key.

20.4) Bob can find S and forge messages using the integrity code once he knows $d(S)_{Bob}$ & Alice. Alice's signature is needed to prove S came from her, while encryption with Bob's public key ensures only Bob can access S.

20.5) Upon receiving the message, both Bob and Ted know K and its encrypted version, allowing either of them to forge a message to the other, pretending to be Alice.

20.7) Although plausible deniability is maintained, the method is insecure. If Alice has sent a message using $[S]^{Alice}$ before, Trudy can use S to create a message for Bob and deceive him into believing it came from Alice.

20.10) To prove a purchase order's legitimacy, the notarization date (n) must be earlier than the minimum of the certificate's expiration date (x) and the CRL date (r). The judge may also require notarization dates for the certificate and CRL to be not long after r.

20.11) If an eavesdropper can guess a message's plaintext, they can confirm their guess by computing the message digest, applying Alice's public key to the signed digest, and checking if it matches the computed message digest.