ISA562_Assignment 5 Mithilaesh Jayakumar(G01206238)

Chapter 11

Qn 3

1. A ⊕ R – It is not secure because an eavesdropper who discovers it , will also discover A.

2. {R + A} A -- It is very secure because we add the Secret of Alice which maintains the integrity of the session key. Also R is not directly shared.

3. {A} A – It is not secure because it is the same for all the sessions and if the intruder figures out Alice's secret then he can impersonate.

4. {R} R+A – It is very secure because R is encrypted with the sum of R and A in addition to the security measures implemented in {R + A} A.

Qn 5

It is not secure as we are sending the challenge in the network layer and then encrypting the challenge with the shared secret key. An eavesdropper can impersonate Bob to Alice through a replay attack. The eavesdropper would send Alice R and Alice would return the encrypted KAlice-bob {R}. Now Bob will send R and the eavesdropper can send back the correct KAlice-bob {R}.

Qn 6

Though we are encrypting R with the private key of Bob and again with the shared secret key, this is also insecure because an eavesdropper can impersonate Bob to Alice and send Alice R plus R encrypted with a key that only he knows, KEve{R}. Alice will send back KEve{R} thinking that it is KBob{R} and KAlice-bob {R} to the eavesdropper. The eavesdropper can now impersonate Alice to Bob and send the correct KAlice-bob {R}.

Qn 13

MD5(KAlice-Bob V R) is not secure because an intruder can impersonate by sending a sequence of challenges R repeatedly for some n number of times to Alice and figure out KAlice-Bob. Though if R is small still it can be compromised. If we use MD5(KAlice-Bob ⊕ R) it is comparatively secure as XOR flips the bits.

Chapter 12

Qn 3

It works if hashn does a hash of all 128-bits of hashn-1. We will generally pad the 64-bit message to get a 128-bit message in the hash function. After doing the hashn we can reduce the 128-bit message to again 64-bit message by discarding the padded 64-bits.

Qn 10

An intruder who has captured Bob's database can use that to try different passwords and compute their W key from that password which can be used to decrypt g^a mod p.

Now the intruder can impersonate as Alice, by calculating b and challenging c which cannot be found until the third step. By now the intruder can figure out the secret keys and the hash functions of both the server and the client resulting in the breach of the data.

Qn 12

In protocol 12.4, we choose a W, a weak secret which is the hash of Alice's password and then Y which is Alice's secret key encrypted with a hash function of her password that is a different hash from the password W. Now if an intruder figures out W, he can only figure out a but not b as {g^ab mod p}Y is encrypted with a different hash function.

In Protocol 12.2, if the intruder has W then he can figure out g^a mod p. By doing so he can also figure out g^b mod p and C1 as he has W. As the intruder now has K he can also figure out C2 causing a vulnerability.

By encrypting g^b mod p and the challenge C1 with a different hash function other than W and signing with sender's private key we verify the sender. Though this helps to overcome the previous vulnerability still it has security drawbacks which is negligible.

Qn 7

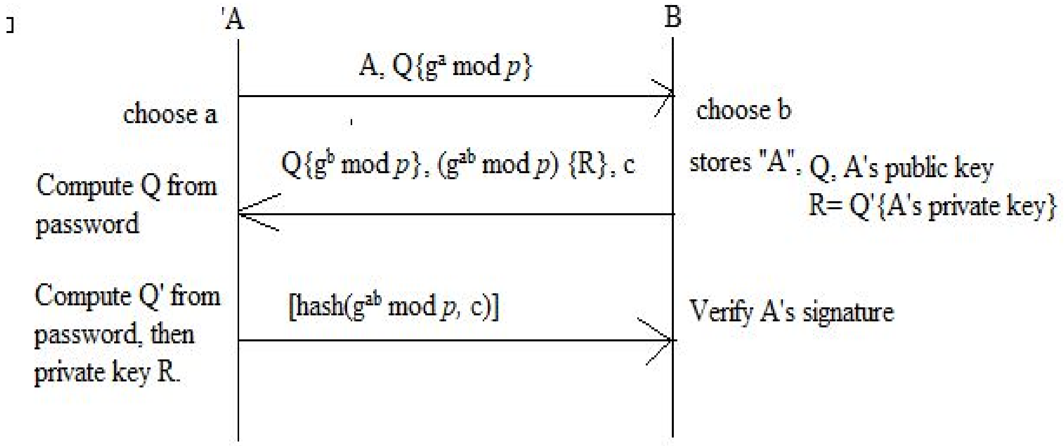
The augmented EKE is the strong password protocol with additional features of security. This additional security was used to prevent the server database from impersonating the user. Impersonating the user means if any person say Alice does the dictionary attack on the server database and Alice is able to find out the password of the user. But with the help of the augmented form of EKE, this attack will be unsuccessful and would not be able to impersonate the user. All the basic schemes such as EKE, SPEKE can be used as an augmented form. The augmented form of EKE is complex in nature because it consumes too many messages.

The working of augmented EKE, SPEKE is as follows:

- The database server will store the prime number derived from the user's password and name it as p.
- Then, the server also saves the value of 2^Q mod p, where Q is the hash of the user's password.

Step 2

The exchange of information will work as follows:


'A
B
A, Q{g^a mod p}
choose a
choose b
Q{g^b mod p}, (g^{ab} mod p) {R}, c
stores "A", Q, A's public key
R= Q'{A's private key}
Compute Q from password
Compute Q' from password, then private key R.
[hash(g^{ab} mod p, c)]
Verify A's signature

Step 3

The figure shows that B can store the private key of A that is R with a function of B's password.

B also stores A's RSA public key which is corresponding to the private key.

In the first message, A sends the first EKE message that contains Diffie Hellman value encrypted in Q.

In the second message, B sends his Diffie-Hellman value with A’s encrypted private key R. that is encrypted with the agreed upon Diffie Hellman key.

A will extract R by decrypting the value of g^ab mod p , and then decrypt R with his own password, to obtain the original private key.

In the third message, A signs a hash of the Diffie-Hellman key and challenge c, and B then verifies the signature of A using the public key stored.

This will help in achieving the augmented property of schemes EKE and SPEKE.

Qn 14

Intruder can do a dictionary attack by impersonating Bob. The Intruder can take advantage of Alice by sending g^a mod p which is less than p and by trying a brute force attack. This can be made secure to verify the identity by adding another round of key exchange after the step 2.

Qn 15

Alice can compute K as she knows a, b, g^b mod p, g^w mod p, P. She can raise (g^b mod p) and (g^w mod p) to the power of a mod p resulting in (g^ab mod p, g^wb mod p) = K

Bob can compute K as he knows b, g^b mod p, g^w mod p, P. He can raise (g^a mod p) and (g^w mod p) to the power of a mod p resulting in (g^ab mod p, g^2wb mod p) = K

It is not secure because the intruder can guess the password and decrypt it. Depending upon the value of p, the intruder could possibly find the correct password in less amount of time.