README

HW 2 Programming A

Needham-Shroeder Symmetric Key Protocol

1) The assumption for this is that the KDC is secure. We also assume that the KDC has already generated the secret keys for Alice and Bob using step 2. To protect against replay attacks we modify the Needham-Shroeder protocol slightly. Alice then sends a message to Bob, and Bob will reply with Alice's identifier and a modified Nonce B, both of which are encrypted by Kbs. Alice then sends this message to KDC containing her identifier, Bob's identifier, Her Nonce A and the encrypted identifier A, and the encrypted modified Nonce B. The server then sends Alice a List of values containing the Nonce that Alice sent, a shared key given by Kab which is generated by the KDC for the expressed usage between Alice and Bob, Bob's identifier given by the String B, and a list containing the Kab, the String A, and the modified Nonce B, which is encrypted by the server using Bob's symmetric key Kbs. This whole list is encrypted using Kas sent using our DES from Hw1 before being sent by the KDC to Alice. Alice will then decrypt all the information, using the Nonce to ensure that the request is correct, verify who it's from, and then prepares to send the last part containing (the symmetric session Key, and Alice's identifier) encrypted via Kbs to Bob. Bob then decrypts it using his Kbs and then is able to verify the sender and now has the symmetric shared key Kab. Bob then sends a message to Alice containing a Nonce B encrypted using the shared key. This is used to confirm that Bob has received the correct symmetric shared key. Alice finally replies to Bob sending a modified Nonce B back to Bob encrypted using their shared key. The modified Nonce is operated upon using a predetermined calculation ex. (Nonce B - 1, Nonce B * 12, etc…). At this point Alice and Bob can now use their symmetric Key Kab to encrypt communications with relative security.

2) Diffie-Hellman Key Exchange using Key Distribution Center

The biggest assumption for this protocol is that the KDC is actually secure. If the KDC is insecure then Ka and Kb will be exposed thus making this key exchange useless. This step will actually occur before the N-S protocol. The way this works is that Alice, Bob, and the KDC already know G and P, which were hard coded for this assignment. We can hard code it because this information is public and is also passed unencrypted. I used G = 9 and P and 23 for this, but you can use a different G and P pairing ex. (9, 23). When Alice starts they must randomly select a secret value A and generate a number given by the equation G^A mod P = X. This value is then passed to the KDC. The KDC then selects its own private key B and then generates a value given by G^B mod P = Y. Alice and the KDC then exchange X and Y and then can solve for their secret key. Ka = y^A mod P and the second secret key is generated by Kb = X^b mod P. This resultant key is now Kas, known only to Alice and the KDC. This process is then repeated between Bob and the KDC to obtain Kbs.