Cryptography & Network Security I - Fall 2018

Homework 2 Theory Part C

Charles Schmitter

October 15, 2018

Q1.

(a) A's public key Y_A is: $Y_A = \alpha^{X_A} \% q = 7^5 \% 71 = 35$
(b) B's public key Y_B is: $Y_B = \alpha^{X_B} \% q = 7^1 2 \% 71 = 13$
(c) The shared secret key is by taking one user's public key and the other user's private key in the formula: $s = Y_A^{X_B} \% q$. We can use both public keys and both private keys in this way to find that the secret key is 34.
(d) If the participants sent x^a mod q instead, then it would be significantly easier for an attacker to compute the secret keys of the users since it is much more difficult to compute the discrete logarithm rather than to simply solve for an unknown base number raised to some power.

Q2.

(a) If an attacker is able to capture a valid signature between an exchange, the attacker can then use the *Birthday Attack*, assuming signature hash collisions exist. Through this type of attack, an attacker can generate many variations of a fraudulent message, and then find a variation that has a signature matching the valid signature found at first. Finally, the attacker can send the fraudulent message with the valid signature.
(b) An attack needs $2^{M/2}$ memory space for an M-bit message.
(c) Due to the nature of the *Birthday Attack*, we know that an M-bit message can be broken in $2^{M/2}$ time. Through a brute force method, we could try each variation of a valid message with each variation of a fraudulent message, leading to $(2^{M/2})^2$ time, which simplifies to 2^M time. We can divide this by the rate given to us ($2^2 0$ hashes per second), and find that it will take 2^{M-20} seconds.
(d) With a 128-bit hash, we can simply plug the bit count in for M from the previous problems. So, (b) will then be $2^6 4$ space, and (c) will then be
2^{44} seconds.

Q3.

I completed this problem assuming we are to use the *Merkle-Hellman Knapsack Cryptosystem* as the *Trapdoor Oneway Function*.

We will begin by forming the β sequence where $\beta_i = a * S_i \bmod p$. So,
$\beta = \{1097, 1175, 1409, 1877, 1009, 1194, 779, 456\}$. Now we can encrypt the plaintext. The ciphertext becomes the sum of each bit multiplied by the corresponding entry in the β sequence. So,

$C = 1097 * 0 + 1175 * 1 + 1409 * 0 + 1877 * 1 + 1009 * 0 + 1194 * 1 + 779 * 1 + 456 * 1 = 5481$. Now, this is sent to the receiver, and the receiver will then decrypt C. Decryption begins with $C' = C * i \mod p$ where i is the multiplicative inverse of $r \mod q$. So, $C' = 1665$. Then, we continue selecting the largest number less than or equal to C' in the given set S, setting C' to that difference each round. We eventually find that we use the following items from S: $\{9, 45, 215, 450, 946\}$. If we select these as 1, and the items that we didn't use as 0, then we find the original plaintext string: 01010111.