**CSCE 4013/5013 Applied Cryptography – Homework 5**

**Due date: November 11, 2018**

**1. (20 points)** Suppose an aggregator wants to know the sum of *n* users’ ages. Design a scheme which allows the aggregator to obtain the sum without revealing any individual user’s age to any other party. Assumptions about the system: users can communicate to each other; no party is trusted; at most two users can collude; no user colludes with the aggregator; no party can eavesdrop the communications between any two users. You must design your own solution. Do not copy from any publication.

**One way to go about getting the sum of all *n* users ages is to have the users give the aggregator their age plus some random number *x*, which the aggregator won’t know. Once the aggregator has the sum, *S*, ask the users to subtract their random number from *S* and once you’ve reached the last user you will have the sum of all user’s ages without having known what that was.**

**2. (16 pts)** Suppose Merkle hash tree is used to authenticate eight messages, m1, m2, …, m8. The eight messages are used to build the Merkle tree, and the root of the tree is signed with a digital signature s.

i) To authenticate m3, besides m3 itself and the signature s of the tree root, which elements should be included in the message? How can the receiver verify the integrity of m3?

**In order to authenticate m3 in this manner you would need the hash value of m4 since its parent is the hash of the hash of m3 and the hash of m4, H(H(m3)+H(m4))**

ii) To authenticate m8, besides m8 itself and the signature s of the tree root, which elements should be included in the message? How can the receiver verify the integrity of m8?

**In order to authenticate m8 in this manner you would need the hash value of m7 since its parent is the hash of the hash of m7 and the hash of m8**

**H(H(m7)+H(m8))**

**3. (12 pts)** Suppose user A is broadcasting packets to n recipients B1, …, Bn. Privacy is not important but integrity is. In other words, each of B1, …, Bn should be assured that the packets he is receiving were sent by A. User A decides to use a MAC.

a. Suppose user A and B1, …, Bn all share a secret key k. User A MACs every packet she sends using k. Each user B­i can then verify the MAC. Explain why this scheme is insecure, namely, show that user B1 is not assured that packets he is receiving are from A.

**Since A and all *Bi* share the same secret key, *k*, then any *Bi* can send messages to anyone in the group that shares *k*. Because of this, B1 can never be sure the message he’s receiving came from A**

b. Suppose user A has a set *S* = {*k1*, …, *km*} of *m* secret keys. Each user *Bi* has some subset ­*Si* of the keys. When A transmits a packet she appends m MACs to it by MACing the packet with each of her *m* keys. When user *Bi* receives a packet he accepts it as valid only if all MAC's corresponding to keys in *Si* are valid. What property should the sets *S1*, …, *Sn* satisfy so that the attack from part (a) does not apply? We are assuming all users *B1*, …, *Bn* are sufficiently far apart so that they cannot collude.

**User Bi can fool *Bj*, assuming *i* ≠ *j*, only if *Bi* has every key *Bj* has. This can be seen when user *Bj* verifies a message by verifying the MACs corresponding to the keys he has. One way to fix this is for every user *Bi, Bj* to have a separate secret key that is not shared.**

c. Show that when *n* = 10 (i.e. ten recipients) the broadcaster A need only append 5 MAC's to every packet to satisfy the condition of part (b). Describe the subsets *S1*, …, *S10* of set {*k1*, …, *k5*} you would use.

**Note that , which means every *Si* only has 2 keys.**

***S1* : {*k1,k2*}**

***S2* : {*k1,k3*}**

***S3* : {*k1,k4*}**

***S4* : {*k1,k5*}**

***S5* : {*k2,k3*}**

***S6* : {*k2,k4*}**

***S7* : {*k2,k5*}**

***S8* : {*k3,k4*}**

***S9* : {*k3,k5*}**

***S10* : {*k4,k5*}**

**4. (12 pts)** In the HORS one-time signature (refer to the slide deck of Module 9 for details), let *t*=16, *l*=128, *k*=4, and let H() be a hash function that creates 16-bit long hash values. Suppose a message m’s hash value is 0110 1001 0011 0010. Then what is this message’s signature? How to verify the signature?

**Start out by generating sixteen 128-bit strings which will form private key *SK* = {*s1, …, s16*}. The public key will be computed as *PK* = {*v1, …, v16*}. Let *h* = *H*(*m*) and split *h* into *k* = 4 substrings of bits each, with each *hj* being interpreted as an integer *ij*:**

***h1* = 01102 = 610**

***h2* = 10012 = 910**

***h3* = 00112 = 310**

***h4* = 00102 = 210**

**The signature will be {s6, *s9, s3­, s2*}**

**In order to verify a signature over message *m*, compute *h*= *H*(*m*) and split *h* into *k* = 4 substrings *h1, h2, h3, h4* of bits each. Interpret each *hj* as an integer *ij* and check if :**

**5. (20 pts)** Revise the Onion Routing protocol presented in the slides of Module 11 to have one new protocol that meets two requirements:

(i) there are only two relays N1 and N2;

**Alice to N1**

**Ee1{k1,N1,Ee2[k2,N2,D|m]}**

**N1 to N2**

**Ee2[k2,D|m]**

**N2 uses its private key to decrypt the final layer and sends message to destination D**

(ii) the source Alice doesn’t want N2 know the content of the request m. Give the request sending and data reply process.