ideas

Donghang Lu

June 2017

1 Base property

1. Homomorphism: suppose C1 = $\{C1_{ij}\}$,a_1,$a_1^{'}$ is for secret S1 (where C1 is commitment matrix and a_1,$a_1^{'}$ is share polynomial for validator node P)and C2 = $\{C2_{ij}\}$,a_2,$a_2^{'}$ is for secret S2. Then C' = $\{C'_{ij}\}$ where $C'_{ij} = C1_{ij}*C2_{ij}$, a = $a_1 + a_2$, a' = $a'_1 + a'_1$ will be for secret S1+S2. (I omit b and b' for simplicity)

2 protocol design

1.First phase: for every validate node P_i, suppose it received mutiple different secret share requests. and now it has (ID.k, send, Ck, a_k,$a_k^{'}$) for each secret with id k. what it will do is to compute C = $\{C_{ij}\}$ where $C_{ij} = \prod Ck_{ij}$ for every k, a = $\sum a_k$ for every k, a' = $\sum a'_k$ for every k. and a set K which contains all different k. and send (ECHO,K,C,a(j),a'(j)) to every validate node P_j.

For example if a node receives sub-share of s1,s2,and s4. it will send (ECHO,K = [1 , 2, 4],C,a(j),a'(j)) to every Pj where C = $\{C_{ij}\}$ where $C_{ij} = C1_{ij}*C2_{ij}*C4_{ij}$, a(j) = a1(j) + a2(j) + a4(j),a'(j) = a1'(j) + a2'(j) + a4'(j). And lets say this (C,a(j),a'(j)) is a [1 ,2 ,4] sub-share.

2.Second Phase: there will be a leader P_l in validate nodes. suppose we want to mix m secrets every time.P_l will have many counters, for each element in the set K of every ECHO message it received, the corresponding counter will add by one.

For example when P_l received an ECHO with set K = [1,2,4]. counter[1], counter[2], counter[4] will add by one.

When there are m counters with value no less than 2t+1. P_l will broadcast this m ids to every other nodes.This means our protocol choose these m secrets to mix.

For example, when m = 3 and P_l finds out that values of counter[1], counter[2], counter[3] are no less than 2t+1, P_l will send (CONFIRM_SET,K = [1,2,3]) to every P_j.

3.Third phase, for every node P_i with its set K, it will receive a K' from P_l in the second phase. suppose K = [1, 2, 4] and K' = [1 ,2, 3]. what P_i is going to do is to change its [1, 2, 4] sub-share to [1, 2, 3] sub-share. Suppose (C,a,a') is P_i's [1 , 2, 4] sub-share, First it can compute C* = $\{C*_{ij}\}$ where $C*_{ij}$ =

$C_{ij}/C4_{ij}$, a* = a - a4, a'* = a' - a4'. and this (C*,a*,a'*) is a [1 ,2] sub-share. next step is to change it to [1 , 2, 3] sub-share. Unfortunately P_i has no info about secret s3. So what it does is to send (HELP,3) to every other nodes. other nodes which has info about s3 will send (OFFER, 3, C,a(i),a'(i)) to P_i. To make sure P_i receives correct information, P_i will wait until it receives 2t+1 OFFER message and be able to find out which message is right. Since there must be t+1 of message with the same C and this C will be the correct one. P_i will use polynomial interpolation to calculate its own a and a' for s3(even if client3 didn't send anything to P_i). Then P_i can get its [1, 2, 3] sub-share. In this way every honest node can finally get its [1 ,2 ,3] sub-share.

4.Forth phase, it is the same as READY phase of normal VSS. The only difference is that all things in (READY,C,a(j),a'(j)) contains info of s1+s2+s3, not just one secret.