problem detected

- information leak in "Step 3: Finding the most likely song index"
- Bob can find out which song Alice is querying in his database
- flaw: Alice only generates a single random number in Step 3

description of the attack

- $z_l = a_l + b_l$: maximum of the cross-correlation between Alice’s snippet and Bob’s database, $l$: index corresponding to Alice’s song
- Bob receives from Alice $c_k = \pi(a_k + b_k) + r = \pi(z_k) + r$, for all $k$
- Bob computes $c_l = \max_k(\pi(z_k) + r) = \pi(z_l) + r$ and $l' = \pi(l)$ in the plaintext and sends the result to Alice

- Bob computes differences over all the $c_k$’s, i.e. $c_2 - c_1 = \pi(z_2 - z_1)$, $c_3 - c_2 = \pi(z_3 - z_2)$, ..., thus removing the random number $r$ (complexity: $O(N)$)
- Bob computes $z_{uvw}$, the maximum of the cross-correlation between snippet $u$ of song $v$ and song $w$ (complexity: $O(N^3)$)
- Bob iteratively tries to find $u_1,v_1,w_1$ and $u_2,v_2,w_2$ such that $z_{u_1v_1w_1} - z_{u_2v_2w_2} \approx \pi(z_i - z_j)$, for all $i,j$ (complexity: $O(N^3)$)
- once Bob completes this, he can reverse the permutation and compute $l = \pi^{-1}(\pi(l))$
- because Bob has access to all $c_k$’s at the same time, he only needs to find some swaps of the permutation in order to get a good idea on what indexes $u,v$ and $w$ he needs to complete the process

solution 1

- instead of generating a single random number and sending all the $c_k$’s to Bob at the same time, Alice can generate a different random number for each pair of $c_k$’s, and send these pairs to Bob one at a time
– attack:
  * Bob can still compute differences over all $c_k$’s and perform the previous attack
  * because Bob does not know which $c_k$’s Alice gives him on each iteration, he has to compute all the possible cross-correlations $z_{uvw}$ in order to invert the permutation

• solution 2
  – Alice can obtain the most likely song index in a secure way using the secure greater-than once again
  – if the secure-than is used in the same way as in Step 2, Bob can still find out which song Alice has, as she must specify the indexes of the values for each comparison, so some changes need to be made

– expected use: $N - 1$ secure greater-than comparisons
  * Bob computes $sGT(z_1, z_2), sGT(z_{12}, z_3), \ldots, sGT(z_{1..(N/2-1)}, z_{N/2}), \ldots, sGT(z_{N/2}, z_{N/2+1})$ and finally $sGT(z_{1..(N/2)}, z_{(N/2+2)..N})$, with $z_{i..j}$ the maximum of the values between $z_i$ and $z_j$
  * at the end of the process, Bob knows that Alice has one of two specific songs, but he has a 1/2 probability of guessing correctly which one
  * $T_{Step3} = \frac{1}{\alpha} \cdot T_{Step2}, \alpha = \#$ cross-correlations per song

– full redundancy use: $\frac{N(N-1)}{2}$ secure greater-than comparisons
  * Bob computes $sGT(z_i, z_j)$ for all possible $i$ and $j$
  * at the end of the process, Bob has no idea on what song Alice has
  * $T_{Step3} = \frac{N}{2\alpha} \cdot T_{Step2}$, which is unacceptably large

– trade-off between privacy and performance: $\frac{N}{2} \log(N)$ secure greater-than comparisons
  * Bob computes $\frac{N}{2}$ secure greater-than comparisons in $\log(N)$ steps
* example: $N = 8$
  - assume $z_1 > z_2 > \ldots > z_8$
  - compute $sGT(z_1, z_2), sGT(z_3, z_4), sGT(z_5, z_6), sGT(z_7, z_8)$
  - compute $sGT(z_1, z_3), sGT(z_2, z_4), sGT(z_5, z_7), sGT(z_6, z_8)$
  - compute $sGT(z_1, z_5), sGT(z_2, z_6), sGT(z_3, z_7), sGT(z_4, z_8)$
* at the end of the process, Bob cannot obtain enough information to tell which song Alice has with more than $1/N$ probability
* $T_{\text{Step3}} \approx \frac{\log(N)}{2\alpha} \cdot T_{\text{Step2}}$, which is smaller than $T_{\text{Step2}}$