6.876/18.426: Advanced Cryptography March 19, 2003

Lecture 12: Improved Non-Interactive Zero-Knowledge

Scribed by: Peng Xie

1 Preface

Today, we have two goals. First, we will construct a NIZK proof system under the assumption of trapdoor permutations. Secondly, we will construct a NIZK proof system that can handle multiple provers with a single random string.

2 Envelope-Based Proof

First of all, we will use an "envelope-based" proof system to demonstrate the intuition of our NIZK proof for the language HAM. We give the definition of HAM as follows:

$$HAM = \{G : \text{ there exists a Hamiltonian tour in} G\}$$

It is known that HAM is NP-complete. We use an adjacency matrix A to denote a directed graph G. The entry $A(i,j) = 1$ means that there is an edge from node i to node j. In the following proof system, there are three parties, prover P, verifier V and the third trusted party T. The proof system works as follows. The input is a graph G with n nodes . T will generate a graph H such that it is a random Hamiltonian tour on n nodes, which we call a *unicycle graph*. However, T puts envelopes on every entry in H and shows the enveloped matrix H to both P and V. Then T whispers the contents of H to P such that P knows the Hamiltonian tour in H. Then P and V works as follows.

P: on (G,H)

1. Find a permutation π such that H is one of the Hamiltonian tours in $\pi(G)$.
2. "Overlap" H with $\pi(G)$
3. Send π and open all the envelopes for the entries in H such that the corresponding entries in $\pi(G)$ are $0's$.

V: on (G,H)

Checks if the opened envelopes are complete and are all $0's$, if so, accepts, otherwise, rejects.

It is easy to see that this proof is complete. Indeed, if the prover P does know $G \in HAM$, then prover can find the permutation π such that H is one of Hamiltonian tours in $\pi(G)$. Soundness also holds because the H is assured to be a unicycle graph even though all the entries of H have been covered by the envelopes. A malicious prover can't do anything

about it. So, if $G \notin HAM$, the malicious prover can't show that all the opened envelopes are $0's$.

This proof system is ZK also. A PPT S can simulate provers by generating a graph H' such that all the entries are 0s, giving a permutation σ to the verifier and revealing all the '0' entries in $\sigma(G)$ to the verifier. All the entries except revealed ones are covered by envelopes, so no one can distinguish it from the real prover. We will use this intuition to construct the NIZK proof we will discuss today.

3 Non-Interactive Zero-Knowledge Proofs

In order to get the NIZK proof from the above envelope-based proof, we need to get a guarantee for the existence of unicycle graph, H when we generate graphs at random. The answer lies in the random string in the NIZK proof. The idea is as follows. With a long enough random string σ, a one-way permutation, and a hard-core predicate of that one-way permutation, we can encode σ into a bunch of random graphs, and with high probability we'll get at least one unicycle graph. A one-way permutation is a function which can be computed in polynomial time, but the inverse is hard to compute. A hard-core predicate of an one-way permutation f is a predicate $B : \{0,1\}^* :\longrightarrow \{0,1\}$, which can be computed by polynomial time, but given $f(x)$, no PPT algorithm can guess $B(x)$ with a nonnegligible advantage. In the following part, we use the graph and matrix interchangeably.

P first partitions the random input string σ into equal size segments, and applies the inverse one-way permutation f^{-1} to map these segments into new segments, then uses the hard-core predicate of f, B to transfer these new segments into string of 1s and 0s, which specifies the adjacency matrix and thus the graph. The "envelopes" are the portions of the reference string σ. They, in a sense, commit to the graphs, without revealing them. Now, the problem is how to map these envelopes into graphs and make sure of the existence of a unicycle graph with overwhelming probability. Naively, we can group $n \times n$ envelops as a n-node graph. But the probability of the existence of unicycle graph is exponentially low. Therefore, we need new idea to encode the envelopes.

3.1 Encoding The Envelopes

Let's consider the following encoding method. If we combine the $\log n^3$ envelopes into one entry in the matrix, that means, if all the consecutive $\log n^3$ envelopes are $1's$, the corresponding entry is 1, otherwise, 0. Then the probability that an entry in the adjacent matrix is 1 is $\frac{1}{2^{\log n^3}}=\frac{1}{n^3}$. Now in order to get the expected number of '1' entries in the graph to be n, we need a matrix of size $n^2 \times n^2$. That is, the number of $1's$ in matrix is expected $n^4 \times \frac{1}{n^3} = n$. Under this probability, for any matrix, the probability that there are exactly n $1's$ entries is, $\binom{n^4}{n}(\frac{1}{n^3})(1-\frac{1}{n^3})^{n^4-n} > \frac{1}{4\sqrt{n}}$, provided that n is large enough. Now, we have $n^2 \times n^2$ matrix and the probability that there are exactly n '1' entries is greater than $\frac{1}{4\sqrt{n}}$, if n is large enough. What's the probability that there is at most one 1 in each column and row given that there are exactly n '1' entries in the matrix? The answer is a constant. This is can be explained by the following example. Let's say, there are $n^2 \times n^2$ drawers arranged in n^2 rows and n^2 columns, and you throw n socks randomly into one of these drawers.

The probability that there is only one sock in each row and column is constant by birthday paradox. We call a "valid configuration" the configuration that there are exactly n 1s in the matrix and all these 1s are arranged in a way such that there is only one 1 in each column and row. The probability that we get a valid configuration is, $\Omega(\frac{c}{\sqrt{n}})$, where c is a constant. We are interested in some special configurations, by which we define to be the configuration that all the nodes in the graph are connected in one cycle. We claims that the fraction of these special configurations in valid configurations is $\frac{1}{n}$. This is explained as the following example. Suppose we have 5 elements. The chance of a configuration is special is $\frac{4}{5}\frac{3}{4}\frac{2}{3}\frac{1}{2} = \frac{1}{5}$. Generalizing this, we get the probability $\frac{1}{n}$. Therefore, we can conclude that the probability that a matrix have a *unicycle graph* is $\Omega(cn^{-\frac{3}{2}})$, where c is a constant. Thus, if the length of the random string σ is long enough, in other words, if we generate enough graphs, we can get the probability that there exist at least one matrix that corresponds to a unicycle graph to be $1 - \Omega(\frac{1}{c^n})$, where c is a constant.

3.2 NIZK Under One-Way Permutation Assumption

Once the prover gets the random string σ and mapping the string into a sequence of graphs, we can use the envelope-based method for finishing the NIZK proof. By the above argument, with high probability, at least one of these graphs is a unicycle graph. The prover will reveal all the entries for the graphs which are not unicycle graphs. For the unicycle graphs, the prover will reveal the $n^2 - n$ rows and $n^2 - n$ columns which contain only $0's$. Then the remaining matrix is unicycle graph. The prover then uses the envelope-based proof system to prove $G \in HAM$. We give the formal description as follows.

P: on (G, σ)

1. Partition the σ into sequence of segments, $u_1, u_2, \ldots, u_m$, where $|u_1| = |u_2| = \ldots = |u_m| = k$.
2. Compute $f^{-1}(u_i)$, where f^{-1} is the inverse of one-way permutation.
3. Compute $B(f_{-1}(u_i))$, get string of envelopes, where B is the hard-core predicate for f.
4. Encode the string of envelopes into graphs.
5. Reveal the non-unicycle graphs by sending $f^{-1}(u_i)$ and envelopes for all such graphs. For all unicycle graphs, reveal the $n^2 - n$ rows and $n^2 - n$ columns which contain only $0's$.
6. Continues the envelope-based proof for each unicycle graph.

V: on (G, σ)

Checks if the revealed matrixes are non-unicycle graphs by computing $f(f^{-1}(u_i))$ and checking if the matrices define non-unicycle graphs. Check if the partially revealed entries are all 0's. Then runs envelope-based protocol verifier.

Completeness holds: if the input graph G has a Hamiltonian tour, then the prover can prove $G \in HAM$. Soundness holds as well: since the input string is long enough, we get overwhelming probability that there exists a unicycle graph. A dishonest prover

can't abandon the unicycle graphs because the prover has to open all the envelopes in the matrix to prove that there is no Hamiltonian tour. Therefore, any prover has to use the unicycle graph in his proof, thus, can't cheat without being caught by the verifier. Now, we need to prove zero-knowledgeness. The problem is that the poly-time S can't compute the inverse of one-way permutation and can't compute the Hamiltonian tour for a given graph G. S can generate a random string, $u_1, u_2, \ldots, u_m$, then compute $B(u_i)$ as envelopes and encodes them into graphs. For the unicycle graphs, S still uses these graphs, but S has to do extra work here. For all entries that are $1's$ in unicycle graph, S randomly re-generates a new random segment until all these entries are $0's$. This is can be done in polynomial time. After doing this, S pretends these graphs are unicycle graphs and computes $f(u_1), f(u_2), \ldots, f(u_m)$ as the input random string σ. S then can simulate by showing that all the revealed entries in the alleged unicycle graph are 0s. By using the hybrid argument, we can show that the output of S is indistinguishable from the view of any provers. The more detail proof can been found in [FLS].

3.3 NIZK Under Trapdoor Assumption

In the above NIZK proof, the prover's power is unlimited because the prover must compute the inverse of one-way permutation. We now construct NIZK proof for polynomial-time prover. This polynomial prover NIZK is based on the assumption of the existence of the family of trap-door permutations. The scheme is similar to the previous one. The prover randomly chooses a trap-door permutation from the family of trap-door permutations. Instead of computing the inverse of a one-way permutation, the prover compute the inverse of the trap-door permutation by using the trapdoor information. Then the prover sends the "index" of the trap-door function to the verifier. Then prover uses the same protocol as before. The proof of the completeness is the same as before. The proof of soundness is sightly different. Here is the question, who selects the trap-door permutation? The answer is the prover. Is it possible that a dishonest prover chooses a particular trap-door permutation such that all the graphs generated by the random string σ don't contain Hamiltonian tour, and therefore, the prover can cheat that any graph has Hamiltonian tour without getting caught by the verifier? The answer is that it is almost impossible. From the argument in 3.1, we know that for any input random string σ and fixed trap-door permutation, the probability that all the generated graphs don't have Hamiltonian tour is $\Omega(\frac{1}{c^n})$, where c is a constant. Without losing generality, we suppose this probability is $\frac{1}{2^n}$. If we enlarge the length of σ by k times, the probability is $\frac{1}{2^{kn}}$, where there are 2^k members of the trapdoor permutation family. By the union bound, the probability that ANY trapdoor permutation allows the prover to cheat is at most $\frac{\frac{1}{2^{kn-k}} \leq 1}{2^n}$.

The ZK-ness proof is similar to before except that we have a family of trap-door permutations rather than a fixed one-way permutation.

4 Multi-Prover

Now, we alter our construction so that we can reuse the same string for multiple provers that do not share state. (Our construction from last time would work for multiple provers if they jointly remembered where they were.)

Under the assumption that one-way function exists, any NIZK proof system with polynomial prover is also witness indistinguishable. As we know, the indistinguishability is preserved if polynomially many witness indistinguishable proof systems share the same random string. Here is how we construct multi-prover version. Let $g : \{0,1\}^n \to \{0,1\}^{2n}$ be a pseudorandom generator. On the random input string σ, the prover partitions the string into two parts, τ and σ',where $|\tau| = 2n$. Then the prover will prove $L' = (G \in HAM \vee \tau = g(s))$, where s is some seed, and $|s| = n$. So, s is a witness for the new problem L' also, which is an NP problem. The prover can use polynomial time reduction to reduce this new problem L' into $G' \in HAM$, and the witness s (or a Hamiltonian path in G, of course) can be reduced into corresponding witness w for the $G' \in HAM$. The prover then can send the G and proof for the $G' \in HAM$ based on σ' to the verifier. It is easy to see that completeness holds here. Soundness also holds because the chance that a random string of length $2n$ is a pseudo random string is $\frac{1}{2^{2n}}$. That means, except with negligible probability, $G' \in HAM \iff G \in HAM$. The simulation of (P, V) can be done by replacing τ with a pseudo random string $y = g(s')$, where s' is the randomly selected string, which is indistinguishable from real random string to any polynomial time distinguisher. The detailed proof can be found in [FLS]. The advantage in this proof system is that we can reuse the random string here. Provers can use different trapdoor functions for different graphs, because of the witness indistinguishability, all these proofs can still be simulated by the same simulator S.

References

[FLS] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple Noninteractive Zero Knowledge Proofs Under General Assumptions. SIAM J. COMPUT. Vol.29, No.1, pp.1-28