Massachusetts Institute of Technology
6.857: Network and Computer Security
Professor Ronald L. Rivest

Handout 19
October 30, 1997

Solutions to Problem Set 5

Problem 5-1. Short commitments

First of all, an apology for the difficulty of this problem. What was looked for was some strong argument that made the TA think that this scheme was either no longer trustworthy (and why) or why it is probably still secure.

The standard approach would be to use the Birthday Paradox to observe that we can at least try 2^{80} messages to find a collision. However, since most people consider MD5 and SHA secure (which use approximately the same size hash) this is not really a useful attack.

If we're really lucky, then the only other approach would be to break the basic underlying problem (discrete log) in a manner similar to that used to demonstrate the security of CvHP (see lecture notes). However, this is not easy to prove, since we have not covered the math required for this kind of analysis – we're not just dealing in mod(n) anymore, we have a very strangely constructed sequence of numbers.

Nevertheless, since this is based on a strong hash function, there is strong reason to believe that the system (which use exponentiation of random numbers) is unlikely to have a distinct pattern on the low-order bits. (After all, this gives away information helpful towards coming up with collisions in the standard CvHP scheme). The equations should fairly strongly randomize all the bits, so that selecting just the last 160 is not likely to make the problem much easier than other schemes which yield hashes that are 160 bits long.

Problem 5-2. Zero-knowledge with a more general verifier

Well, all this problem is really asking is to demonstrate that this protocol is zero-knowledge in the face of a different verifier. Since we know that this is a zero-knowledge protocol. We know that ANY verifier can produce a transcript of a conversation that looks like a transcript of a conversation he had with a valid prover. For this problem, we just describe how this verifier does it:

- Flip a coin $b \in_R \{0, 1\}$.
- If $(b = 0)$
  - choose $k \in_R Z_{p-1}$
  - set $s = g^k \pmod{p}$.
  - compute c as the XOR of the last two bits of s.
  - if $c = b$ then write out s, c, k, otherwise, go back to start.
- Else $(b = 1)$

–choose $r \in_R Z_{p-1}$

–set $s = \frac{g^r}{g^x} \pmod{p}$.

–compute c as the XOR of the last two bits of s.

–if $c = b$ then write out s, c, r, otherwise, go back to start.

Informal Analysis: No matter what the distribution of the last two bits are, we claim that this simulation will achieve the same distribution whether talking to a real prover or not. The reason for this is that the simulation picks a random number in both cases: $(r + x)$ is as random as r. It guesses what the challenge will be with equal probability, thereby knocking out half of both the entries with $c = 1$ and $c = 0$, and so leaving the distributions the same.