CS 161 Fall 2008 — Computer Security — Dawn Song — Notes 6

We will consider the following authentication scheme: the user selects a number $N = P \cdot Q$ product of two large primes, and a number $y = x^2 \bmod N$. The server is given N, y and to login the user must prove that she knows $x : x^2 = y \bmod N$. Notice the similarity between this and the RSA encryption function — here we are squaring instead of cubing (when $e = 3$ in RSA) to implement our hard to invert function. Indeed, it turns out that computing square roots modulo N is provably as hard as factoring N.

Before we can state the zero-knowledge protocol and establish its properties, we first state a few facts about numbers which are perfect squares modulo N. Let us restrict our attention to numbers $0 \leq a \leq N-1$ which are relatively prime to N (i.e. $gcd(a,N) = 1$; note that if the gcd is not 1 then it must be P or Q, so such a's are rare and lucky choices that we will not consider). This set of numbers is denoted Z_N^*. For example, for $N = 15$, we would consider the numbers $Z_{15}^* = \{1,2,4,7,8,11,13,14\}$. Among these numbers only 1 and 4 are perfect squares. Each has four square roots, $\{1,4,11,14\}$ and $\{2,7,8,13\}$ respectively. The square roots come in pairs, e.g. $13 = -2 \bmod 15$ and $8 = -7 \bmod 15$. In fact, for general $N = P \cdot Q$, exactly one quarter of the elements of Z_N^* are perfect squares and every perfect square $a \bmod N$ has four square roots $\overset{+}{-} x$ and $\overset{+}{-} y$. Moreover, multiplying a square by a square gives another square, since $x^2 \cdot z^2 \bmod N = (xz)^2 \bmod N$.

The protocol:

The prover knows $x : x^2 = y \bmod N$. She wishes to prove to the verifier that she knows such a value x.

1. The prover picks a random value $r \bmod N$ and computes $s = r^2 \bmod N$ and sends s to the verifier.

2. The verifier randomly flips a coin and sends the prover the coin flip.

3. The prover sends either $t = r \bmod N$ if the coin flip is head or $t = rx \bmod N$ if the coin flip is tail.

4. The verifier checks that the received number matches the challenge. In particular, if the coin flip is head, then $t^2 = s \bmod N$; if the coin flip is tail, then $t^2 = sy \bmod N$.

Let us prove that this protocol provides a zero-knowledge proof of knowledge of a square root of $y \bmod N$. We will show that if the prover does not know a square root of $y \bmod N$ then the honest verifier will catch her cheating with probability at least $1/2$. This will establish that the protocol constitutes a proof of knowledge. And we will show that the verifier cannot extract any extra information from the prover no matter how he deviates from protocol. To do so we will show that for every verifier (no matter how dishonest), there is a simulator that can recreate the verifier's view (his conversation with the prover) without any knowledge of a square root of $y \bmod N$. This will establish that the protocol is zero-knowledge.

Knowledge extractor: (Optional reading)

If the prover wishes not to be caught cheating, she must be able to answer both possible challenges of the verifier. We will argue that such a prover must know a square root of $y \bmod N$. For the purposes of the proof let us assume that there is a hypothetical knowledge extractor who can travel backward in time, and after issuing the first challenge and receiving the answer, the knowledge extractor travels back in time and issues the second challenge. By our assumption the prover can answer both challenges and therefore the knowledge extractor receives $u : u^2 = s \bmod N$ and $v : v^2 = sy \bmod N$. Now $w = v/u \bmod N$ is a square root of $y \bmod N$, since $w^2 = v^2/u^2 = sy/s = y \bmod N$. Thus the knowledge extractor can obtain a square root of $y \bmod N$, therefore establishing that the prover must have known a square root of $y \bmod N$. It is important

to understand that the knowledge extractor is a hyothetical construct. The protocol requires that the prover only answer if the verifier issues one of the two possible challenges. Also note that a dishonest prover can cheat with probability $1/2$. This probability of cheating can be decreased to $1/2^k$ by repeating the protocol k times.

The Simulator: (Optional reading, out of scope)

What is the verifier's view of the protocol. Note that we are now assuming that the prover is honest and that the verifier is trying to trick the prover into revealing information beyond her knowledge of some square root of y mod N. Challenge I by the verifier is s a uniformly random perfect square modulo N. Let us show that the second challenge sy mod N is also a uniformly random perfect square modulo N. To see this first notice that sy is a perfect square, since it is a product of two perfect squares. Also notice that multiplication by y mod N is a permutation of the numbers modulo N. This is because all the numbers we are working with are relatively prime to N, and therefore we can divide by y mod N to show that if $sy = s'y$ mod N then $s = s'$ mod N. Thus multiplication by y is a one-to-one mapping from the set of perfect squares to itself, and is therefore a bijection. Thus if we pick s at random among the perfect squares, then sy mod N is also uniformly random among the perfect squares modulo N.

The simulator selects a random r mod N, and with probability $1/2$ sends the verifier r^2 mod N and with probability $1/2$ sends the verifier r^2/y mod N. If the verifier issues challenge I, then in the first case the simulator responds with r mod N and otherwise rewinds the simulation and starts again. If the verifier issues challenge II, then in the second case, the simulator responds with r mod N, and otherwise it rewinds the simulation and starts again. Since the simulator's choice is independent of the choice of challenge issued by the verifier, it follows that the simulation will succeed in satisying the verifier with probability at least $1/2$. The verifier's view is accurately recreated by the simulator, since it just consists of a challenge consisting of a uniformly random perfect square modulo N, followed by a response from the prover consisting of a square root of this number.