Institut for Matematik og Datalogi
Syddansk Universitet

October 23, 2019
JFB

Cryptology – 2019 – Lecture 12

Announcements

1. All lectures on Thursdays will start at 16:10, instead of 16:15.
2. Lectures on Thursday, October 24, and Thursday, November 21, are cancelled.

Lecture, October 22

We covered subsections 1.3.6 (Shank's Algorithm for computing square roots modulo a prime) and 1.3.9. I presented a proof of correctness for Shank's Algorithm, but discovered an error at the end of the proof of the second invariant, despite the intuition being correct. A correct proof of correctness is presented below. We also went over problems from the first assignment.

Lecture, October 29

We will cover sections 2.1-2.4.

Lecture, November 4

We will finish chapter 15, start on chapter 16, and cover section 3.1.

Problem session November 5

We will finish the problems not finished on October 31. If there is time, I will lecture.

Proof of correctness of Shank's Algorithm (Algorithm 1.3 in the textbook

The algorithm is proven correct by proving two loop invariants for the **while** loop.

Invariant 1: $x^2 \equiv a \cdot b \pmod{p}$

Note that the **while** loop terminates when $b \equiv 1 \pmod{p}$. Thus, if this invariant holds and the loop terminates, the correct answer is returned, since x is returned.

Proof of Invariant 1: The proof is by induction on the number of times through the **while** loop. The base case is just before the loop is executed for the first time. Then $b \equiv a \cdot x^2 \equiv a \cdot (a^{\frac{q-1}{2}})^2 \equiv a^q \pmod{p}$ and x is updated to $a \cdot a^{\frac{q-1}{2}} \pmod{p}$, so $x^2 \equiv a^{q+1} \pmod{p}$ and $x^2 \equiv a \cdot b \pmod{p}$. Thus, the base case holds.

Suppose Invariant 1 holds at the start of an iteration of the **while** loop. Let x' be the new value of x and b' be the new value of b computed in this iteration. Then $x' \equiv x \cdot t \pmod{p}$ and $b' \equiv b \cdot t^2 \pmod{p}$. Thus, $x'^2 \equiv (x \cdot t)^2 \equiv x^2 \cdot t^2 \equiv (a \cdot b)t^2 \equiv a \cdot b'$, where the induction hypothesis is used in the third congruence. Thus, Invariant 1 holds for all iterations.

Invariant 2: $b^{2^{r-1}} \equiv 1 \pmod{p}$ and $y^{2^{r-1}} \equiv -1 \pmod{p}$.

Note that the **while** loop is not entered if $b \equiv 1 \pmod{p}$. Thus, assuming Invariant 2, $r > 1$ at the start, every time the the loop is entered.

Proof of Invariant 1: The proof is by induction on the number of times through the **while** loop. The base case is just before the loop is executed for the first time. Then $2^{r-1} = 2^{e-1}$. Since n is a quadratic nonresidue modulo p, $n^{\frac{p-1}{2}} \equiv -1 \pmod{p}$. Then,

$$y^{2^{r-1}} \equiv (n^q)^{2^{e-1}} \equiv n^{\frac{p-1}{2}} \equiv -1 \pmod{p}.$$

In addition, from the proof of Invariant 1, we know that $b \equiv a^q$ at this point, so $b^{2^{r-1}} \equiv (a^q)^{2^{e-1}} \equiv a^{\frac{p-1}{2}} \equiv 1$, since a is a quadratic residue. Thus, the base case holds. Suppose Invariant 2 holds at the start of an iteration of the **while** loop. Let y' be the new value of y and b' be the new value of b computed in this iteration. Then, $b' \equiv b \cdot t^2 \pmod{p} \equiv b' \cdot y' \pmod{p}$. Note that the value of m is chosen to be the smallest integer such that $b^{2^m} \equiv 1 \pmod{p}$. This means that if $m - 1 \geq 0$ (and otherwise the loop is not executed again, so there is nothing to be proven), then $b^{2^{m-1}} \equiv -1 \pmod{p}$, since it is

square root of 1, but not equal to 1. We know that $m < r$ since inductively $b^{2^{r-1}} \equiv 1 \pmod{p}$. Thus, the calculation $t \equiv y^{2^{r-m-1}} \pmod{p}$ makes sense. Then, $y' \equiv t^2 \equiv y^{2^{r-m}} \pmod{p}$, and $y'^{2^{m-1}} \equiv y^{2^{r-1}} \equiv -1 \pmod{p}$, where the last congruence holds by the induction hypothesis. Since r is updated to have the value m, $y^{2^{r-1}} \equiv -1 \pmod{p}$ holds inductively. Then,

$$b'^{2^{r-1}} \equiv (b \cdot y')^{2^{m-1}} \equiv b^{2^{m-1}} \cdot y'^{2^{m-1}} \equiv (-1) \cdot (-1) \equiv 1 \pmod{p}.$$

Thus, Invariant 2 holds by induction.

This concludes the proof. Note that since r is updated to $m < r$ each time through the **while** loop and $1 \leq r \leq e$ at the start of each iteration, the loop is executed at most $e \leq \log_2(p-1)$ times, and the algorithm is polynomial time. (Remember that fast modulo eponentiation is used everywhere.)