Notes for Recitation 5

1 The Pulverizer

To employ RSA, we’ll need a mathematical tool that dates to sixth-century India, where it was called \textit{kuttak}, “The Pulverizer”. (The Pulverizer is often lamely referred to as the “extended Euclidean GCD algorithm”, but that will not happen in \textit{this} recitation.) The Pulverizer constructively proves the following theorem:

\textbf{Theorem 1.} For every pair of positive integers $a$ and $b$, there exist integers $x$ and $y$ such that:

$$x \cdot a + y \cdot b = \gcd(a, b).$$

For example, if $a = 259$ and $b = 70$, then the Pulverizer works as follows:

<table>
<thead>
<tr>
<th>$a'$</th>
<th>$b'$</th>
<th>$r = a' - q \cdot b'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>259</td>
<td>70</td>
<td>49 = $259 - 3 \cdot 70$</td>
</tr>
<tr>
<td>70</td>
<td>49</td>
<td>21 = $70 - 1 \cdot 49$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$= 70 - 1 \cdot (259 - 3 \cdot 70)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$= -1 \cdot 259 + 4 \cdot 70$</td>
</tr>
<tr>
<td>49</td>
<td>21</td>
<td>7 = $49 - 2 \cdot 70$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$= (259 - 3 \cdot 70) - 2 \cdot (-1 \cdot 259 + 4 \cdot 70)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$= 3 \cdot 259 - 11 \cdot 70$</td>
</tr>
<tr>
<td>21</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

In the first two columns, we’re just carrying out Euclid’s gcd algorithm. We keep track of two variables, $a'$ and $b'$. Initially, we set $a' = a$ and $b' = b$. At each step, we compute $r = a' \ \text{rem} \ b'$. If $r = 0$, then the procedure terminates and $b' = \gcd(a, b)$. If $r \neq 0$, then let $a' = b'$, let $b' = r$, and continue the procedure.

In the third column, we express the remainder $r$ in the form $r = a' - q \cdot b'$ at each step. By replacing $a'$ and $b'$ with previously-computed linear combinations of $a$ and $b$, we can express $r$ as a linear combination of $a$ and $b$ as well. Consequently, the final equation in the third column expresses $\gcd(a, b)$ as a linear combination of $a$ and $b$:

$$3 \cdot 259 - 11 \cdot 70 = \gcd(259, 70) = 7$$
Problem 1. Let’s try out RSA! There is a complete description of the algorithm on the next page. You’ll probably need extra paper. Check your work carefully!

(a) As a team, go through the beforehand steps.

- Choose primes $p$ and $q$ to be relatively small, say in the range 10-20. In practice, $p$ and $q$ might contain several hundred digits, but small numbers are easier to handle with pencil and paper.
- Try $e = 3, 5, 7, \ldots$ until you find something that works. Use Euclid’s algorithm to compute the gcd.
- Find $d$ using the Pulverizer.

When you’re done, put your public key on the board. This lets another team send you a message.

(b) Now send an encrypted message to another team using their public key. Select your message $m$ from the codebook below:

- 2 = Greetings and salutations!
- 3 = Yo, wassup?
- 4 = You guys suck!
- 5 = All your base are belong to us.
- 6 = Someone on our team thinks someone on your team is kinda cute.
- 7 = You are the weakest link. Goodbye.

(c) Decrypt the message sent to you and verify that you received what the other team sent!

(d) Explain how you could read messages encrypted with RSA if you could quickly factor large numbers.

Solution. Suppose you see a public key $(e, n)$. If you can factor $n$ to obtain $p$ and $q$, then you can compute $d$ using the Pulverizer. This gives you the secret key $(d, n)$, and so you can decode messages as well as the intended recipient.
RSA Public-Key Encryption

**Beforehand** The receiver creates a public key and a secret key as follows.

1. Generate two distinct primes, $p$ and $q$.
2. Let $n = pq$.
3. Select an integer $e$ such that $\gcd(e, (p - 1)(q - 1)) = 1$.
   
   The *public key* is the pair $(e, n)$. This should be distributed widely.
4. Compute $d$ such that $de \equiv 1 \pmod{(p - 1)(q - 1)}$.
   
   The *secret key* is the pair $(d, n)$. This should be kept hidden!

**Encoding** The sender encrypts message $m$ to produce $m'$ using the public key:

$$m' = m^e \text{ rem } n.$$

**Decoding** The receiver decrypts message $m'$ back to message $m$ using the secret key:

$$m = (m')^d \text{ rem } n.$$

**Problem 2.** A critical question is whether decrypting an encrypted message always gives back the original message! Mathematically, this amounts to asking whether:

$$m^{de} \equiv m \pmod{pq}.$$

Note that the procedure ensures that $de = 1 + k(p - 1)(q - 1)$ for some integer $k$.

(a) Use Euler’s Theorem to prove that $m^{de} \equiv m \pmod{pq}$ for all messages $m$ relatively prime to $pq$. In practice, is $m$ likely to be relatively prime to $pq$ or not?

**Solution.**

$$m^{de} = m^{1+k(p-1)(q-1)} \quad \text{(by choice of } e)$$

$$= m^{1+k\phi(pq)} \quad \text{since } \phi(pq) = (p-1)(q-1)$$

$$\equiv m^{1+k\phi(pq)} \pmod{pq}$$

$$\equiv m \cdot (m^{\phi(pq)})^k \pmod{pq}$$

$$\equiv m \cdot 1^k \pmod{pq} \quad \text{by Euler’s Theorem}$$

$$\equiv m \pmod{pq}.$$

Only one $p$th of the numbers from 0 to $pq - 1$ are divisible by $p$, and likewise only one $q$th by $q$. So when $p$ and $q$ are hundred-digit primes, the fraction of numbers
relatively prime to both \( p \) and \( q \) is very close to 1, which means it is nearly certain and a random \( m \) will be relatively prime to both \( p \) and \( q \).

**(b)** This congruence actually holds for all messages \( m \). First, use Fermat’s theorem to prove that \( m \equiv m^{de} \pmod{p} \) for all \( m \). (Fermat’s Theorem says that \( a^{p-1} \equiv 1 \pmod{p} \) if \( p \) is a prime that does not divide \( a \).)

**Solution.** If \( m \) is a multiple of \( p \), then the claim holds because both sides are congruent to 0 mod \( p \). Otherwise, suppose that \( m \) is not a multiple of \( p \). Then:

\[
m^{1+k(p-1)(q-1)} \equiv m \cdot (m^{p-1})^{k(q-1)} \pmod{p} \\
\equiv m \cdot 1^{k(q-1)} \pmod{p} \\
\equiv m \pmod{p}
\]

The second step uses Fermat’s theorem, which says that \( m^{p-1} \equiv 1 \pmod{p} \) provided \( m \) is not a multiple of \( p \).

**(c)** By the same argument, you can equally well show that \( m \equiv m^{ed} \pmod{q} \). Show that these two facts together imply that \( m \equiv m^{ed} \pmod{pq} \) for all \( m \).

**Solution.** We know that:

\[
p \mid (m - m^{ed}), \\
q \mid (m - m^{ed}).
\]

Thus, both \( p \) and \( q \) appear in the prime factorization of \( m - m^{ed} \). Therefore, \( pq \mid (m - m^{ed}) \), and so:

\[
m \equiv m^{ed} \pmod{pq}.
\]