Note 7 Supplement: RSA Extras

Computer Science 70
University of California, Berkeley

Summer 2018

1 One-Time Pad

The **exclusive OR (XOR)** $x \oplus y$ of two bits x and y is defined by:

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

In other words, $x \oplus y$ equals 1 if and only if x and y are different bits. Notice that $x \oplus y$ is the same as $x + y \bmod 2$. For any $x \in \{0, 1\}$, we have $x \oplus x = 0$ and $x \oplus 0 = x$. So, for any $y \in \{0, 1\}$, we have $y \oplus x \oplus x = y \oplus 0 = y$.

We can extend the XOR operation to work on bit strings x and y of the same length by applying the XOR operation bitwise.

Example 1. $01000 \oplus 11100 = 10100$.

For bit strings x and y of the same length, we again have $y \oplus x \oplus x = y$. This actually gives us the simplest method to encrypt our messages, known as the **one-time pad**. To send a message m (a bit string), the sender and receiver both agree (in advance) on a secret key k, which is a bit string of the same length as the message. The sender sends $m \oplus k$ to the receiver, and the receiver decrypts the message by $m \oplus k \oplus k = m$.

If an eavesdropper intercepts the encrypted message $m \oplus k$, then without knowledge of the secret key k, the one-time pad is unbreakable. Indeed, since the secret key is unknown, then the eavesdropper must think that any secret

key is possible. Given any message m', then $m' \oplus m \oplus k \oplus m' = m \oplus k$, which means that the encrypted message $m \oplus k$ could have also come from the message m' with the secret key $m \oplus k \oplus m'$. We have just shown that the encrypted message could have come from *any* starting message, which means that the eavesdropper knows nothing about the original message.

The one-time pad is not very convenient, however, because in order to guarantee the safety of the scheme, the secret key should really be discarded after one use (hence the name "one-time pad"). Since the sender and receiver must agree upon the secret key beforehand, the inability to reuse the secret key significantly hinders the practicality of the scheme. Nevertheless, the one-time pad can be useful when combined with other schemes.

2 Application of RSA: Digital Signatures

A signature is meant to provide proof of an individual's identity. In order for the signature to be a valid proof, the signature must have the property that no other individual can produce the same signature. Unfortunately, in the real world, we know that signatures can be forged.

Inspired by this idea, we introduce the concept of a **digital signature**. As before, a digital signature is supposed to provide proof of an individual's identity. However, the property that "no other individual can produce the same signature" is replaced by the property that "no other individual can *reliably* produce the same signature *efficiently*". The idea is that someone who wants to forge the signature must use some brute force method which is computationally infeasible, e.g., would require centuries or more to compute.

Suppose that you have a RSA public key (N, e) with corresponding private key d. One way to provide a "signature" is to reveal your private key d. If we assume that RSA is unbreakable, then the private key cannot be computed efficiently from the public key, so this would indeed constitute a signature. Unfortunately this has the drawback of revealing your private key.

Instead, the signature scheme proceeds as follows. A verifier provides the individual with some randomly chosen $x \in \{0, 1, \ldots, N-1\}$ and asks the individual for $x^d \bmod N$. The verifier can then check that $x^{ed} \equiv x \pmod{N}$.

If the individual knows the private key d, then this computation is fast. However, a forger without knowledge of the private key must labor to find the $y \in \{0, 1, \ldots, N-1\}$ such that $y^e \equiv x \pmod{N}$. If RSA is unbreakable, then this cannot be done efficiently. Presently we believe that you cannot do

meaningfully better than exhaustive search, which can easily take centuries if N is large enough.

The verifier can play this game with the individual multiple times until the verifier is satisfied that the individual is not forging the signature.

3 RSA Attacks

The RSA scheme presented in the notes is known as "textbook RSA". When RSA is used in practice, there are extra bells and whistles that are added to the scheme to improve its security. In this section we describe a couple of known attacks against the RSA scheme.

The first attack warns against using RSA alone. Suppose that you take your credit card m and pass it to the encryption function E to get your encrypted credit card number $E(m)$. The encrypted credit card number $E(m)$ is then sent to a company such as Amazon in order to complete a credit card transaction. However, an eavesdropper sees $E(m)$. The eavesdropper can then send $E(m)$ to the company again in order to make his or her own purchases, effectively stealing your credit card.

The method to prevent this attack is to take your credit card number m, and in each new transaction, *pad* your credit card number with a randomly generated string at the end to form a longer, random string m'. Then, send $E(m')$ to the company. This is called *RSA with padding*. The randomness ensures that even if you send the same message twice, the encrypted messages will most likely differ, so that if the company receives the same encrypted message $E(m)$ twice in a row, then it will know to be suspicious.

The second attack is about unwittingly giving away information. Say that an attacker intercepts the encrypted message $E(m)$. Since the attacker cannot decrypt the message, it asks the company to decrypt the message in a roundabout way. First the attacker picks a random number r, and asks the company to please decrypt the message $E(m) \cdot r^e \bmod N$, where (N, e) is the public key. After multiplying $E(m)$ by r^e, the result is a seemingly innocuous string, so the company complies with the request, sending back the decrypted message mr. Now, since the attacker knows r, he or she also knows $r^{-1} \bmod N$, and using this, the attacker can recover the original message m.

It may be surprising to learn that our cryptosystems (such as RSA) are not *provably* secure, but nevertheless they are used every day.